EECE 5639 Computer Vision I

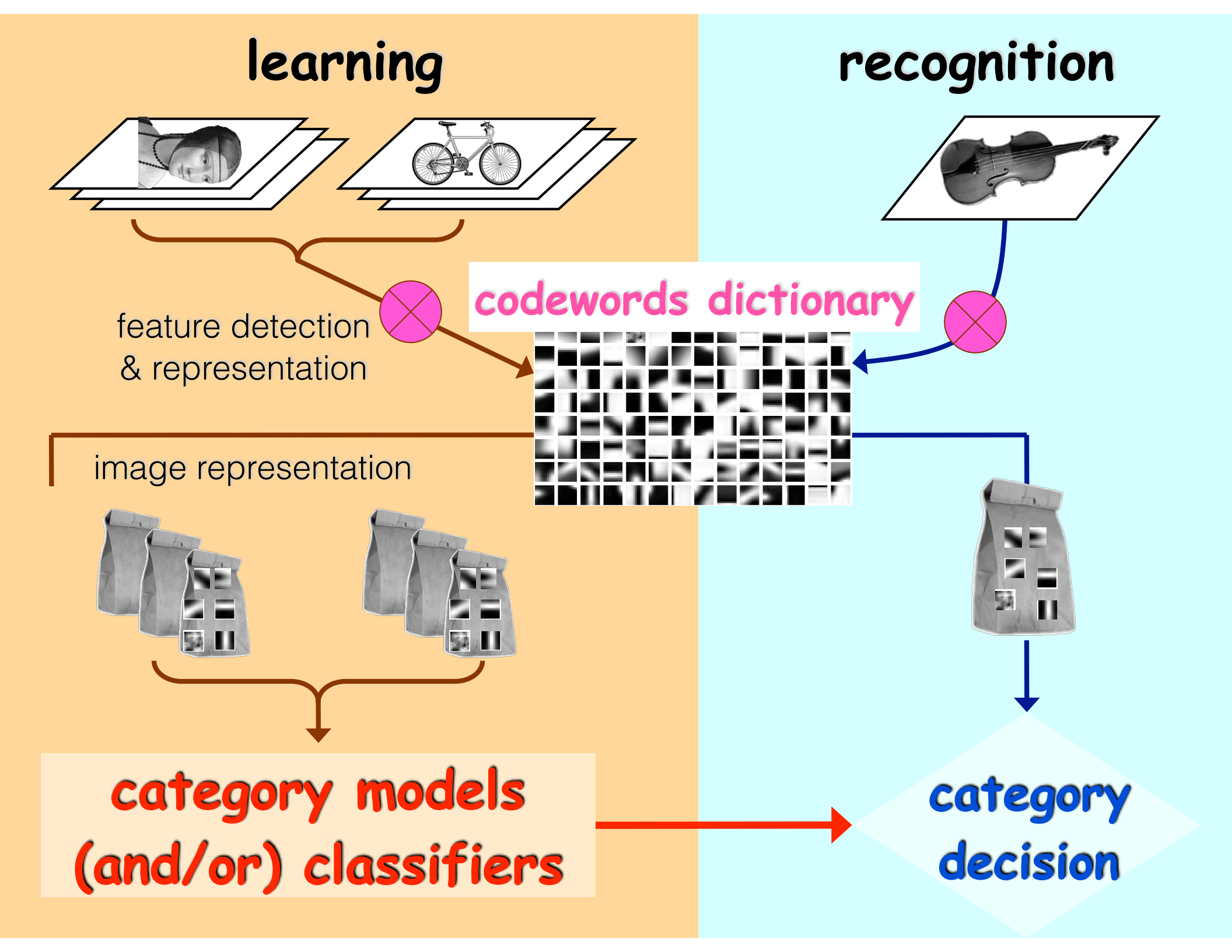
Lecture 23

- **BoW: Implicit Shape Model**
- **Deep Learning**
- **Hw 5 is out. Now Due April 19**

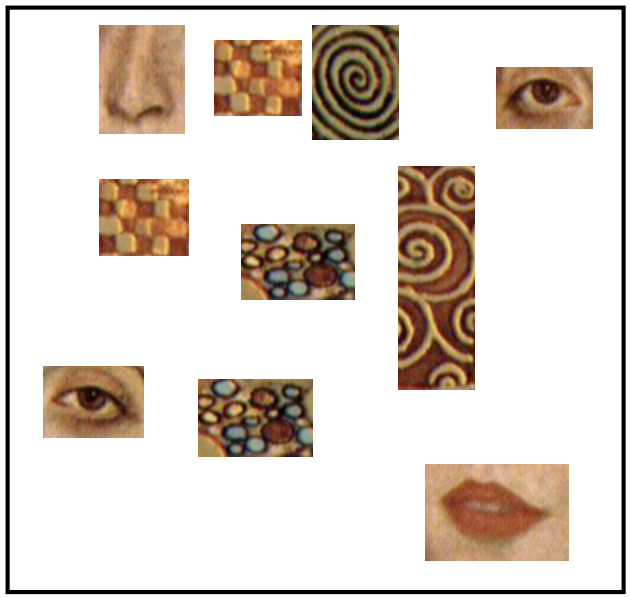
Next Class

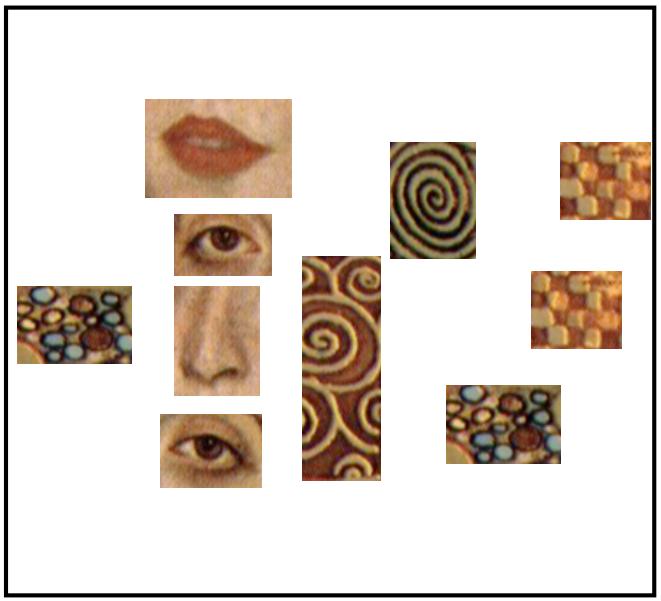
- **Final Review**

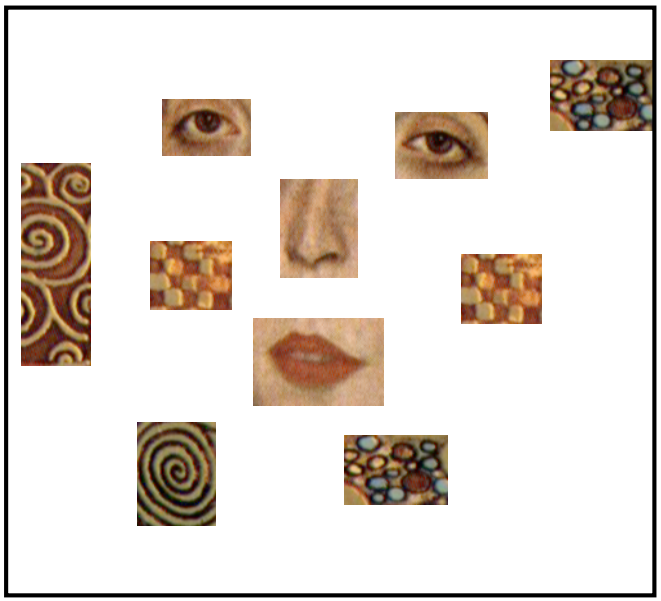
Object → Bag of 'words'


learning
recognition
codewords dictionary
feature detection
& representation
image representation
category models
(and/or) classifiers
category
decision

Problem with bag-of-words







All have equal probability for bag-of-words methods

Location information is important

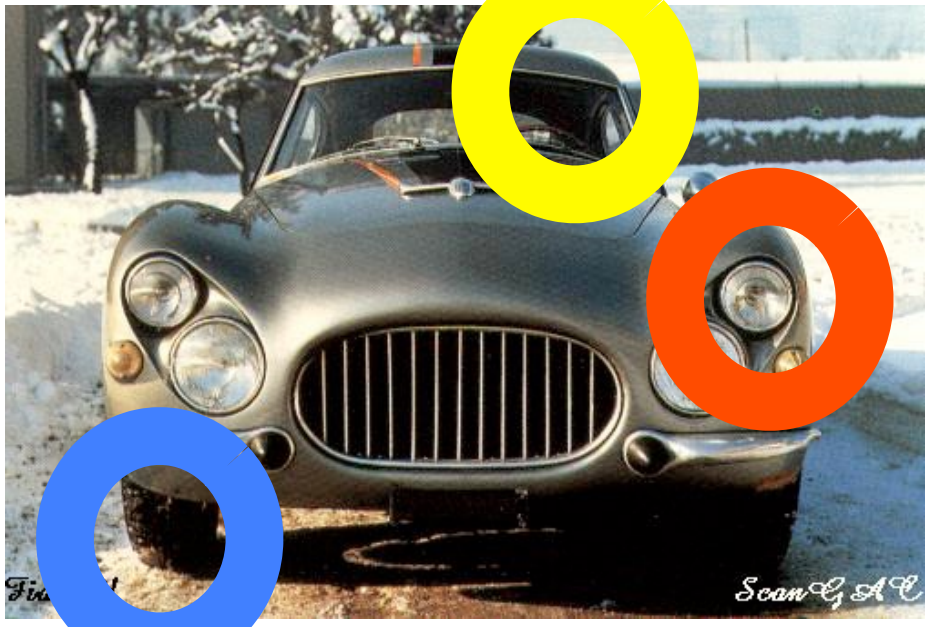
Bag of Words and Spatial Information

A bag of words throws away spatial relationships between features.

Middle ground:

- Visual “phrases”: frequently co-occurring words
- Semi-local features: describe configuration, neighborhood
- Let position be part of each feature
- Count bags of words only within sub-grids of an image
- After matching, verify spatial consistency

Parts & Structure



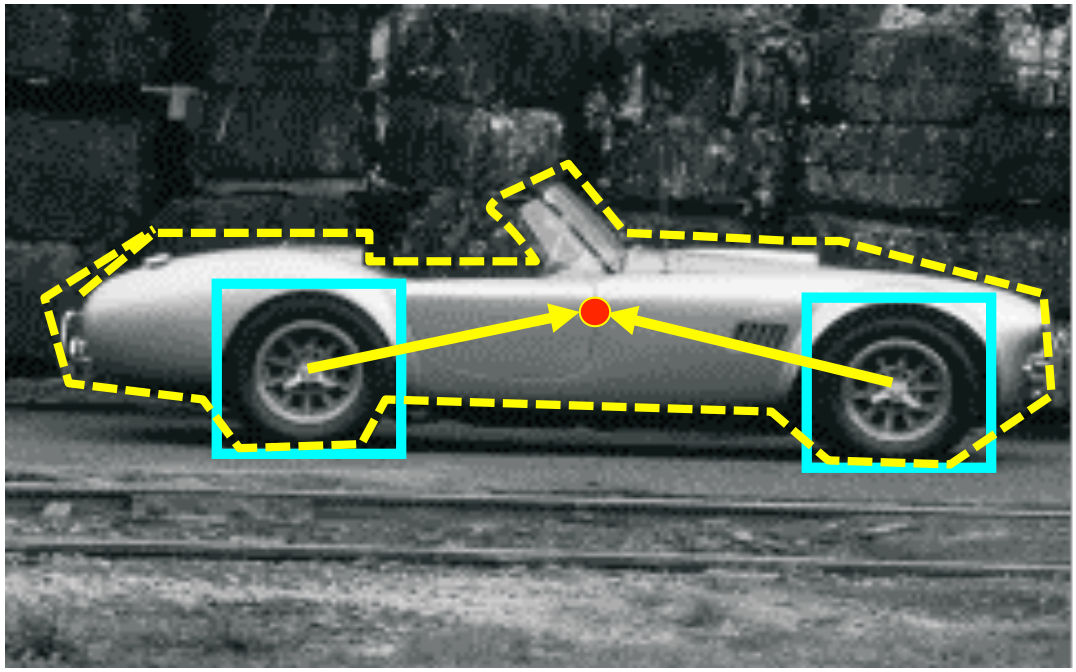
Implicit Shape Model

Use Hough Space Voting to find object:

- Learn spatial distributions of the words wrt to a “reference point”
- Use HT to vote for reference points in the test image

Implicit shape models

- Visual vocabulary is used to index votes for object position
  - [a visual word = “part”]



training image annotated with object localization info

visual codeword with displacement vectors

B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model, ECCV Workshop on Statistical Learning in Computer Vision 2004

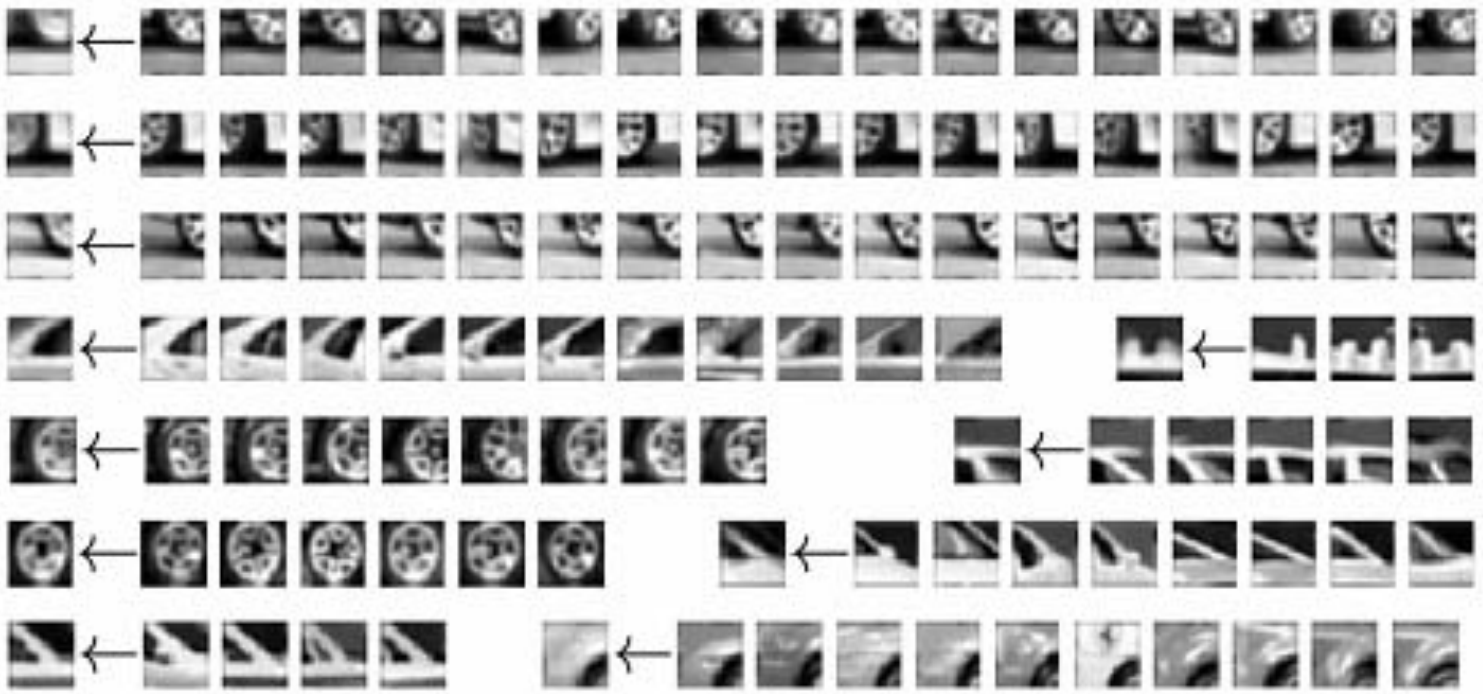
Implicit shape models

- Visual vocabulary is used to index votes for object position
  - [a visual word = “part”]

test image

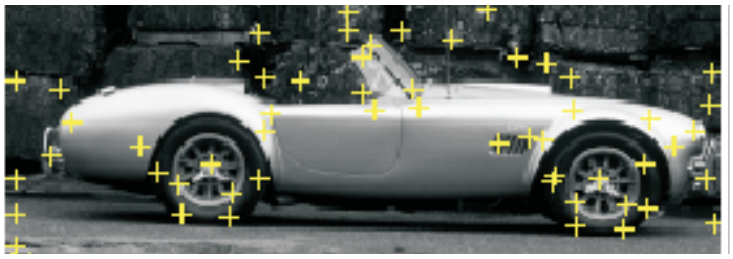
Implicit shape models: Training

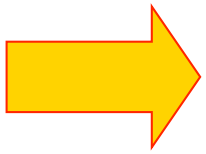
1. Build vocabulary of patches around extracted interest points using clustering

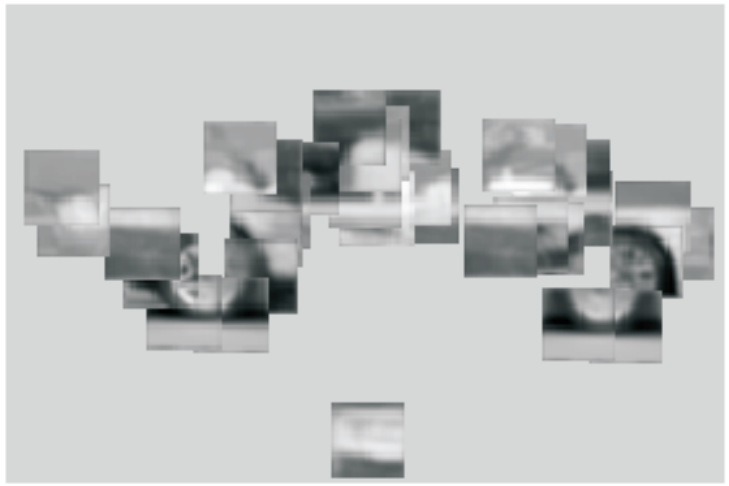


Implicit shape models: Training

1. Build vocabulary of patches around extracted interest points using clustering
2. Map the patch around each interest point to closest word

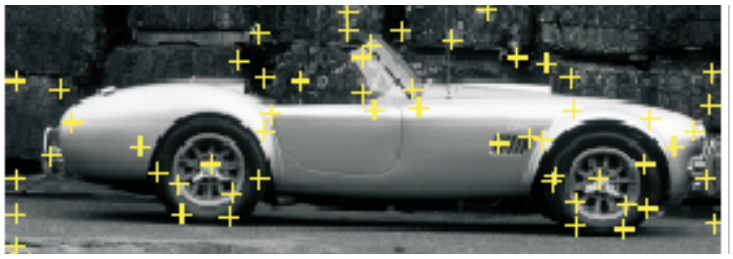


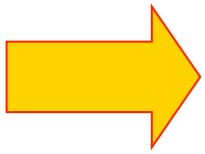


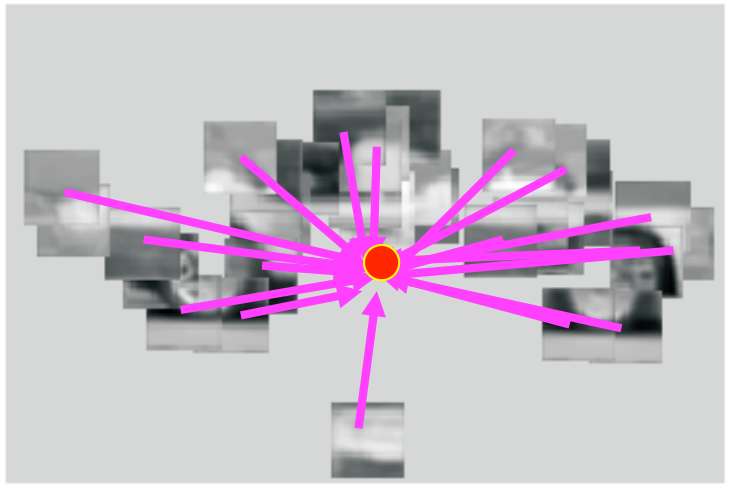


Implicit shape models: Training

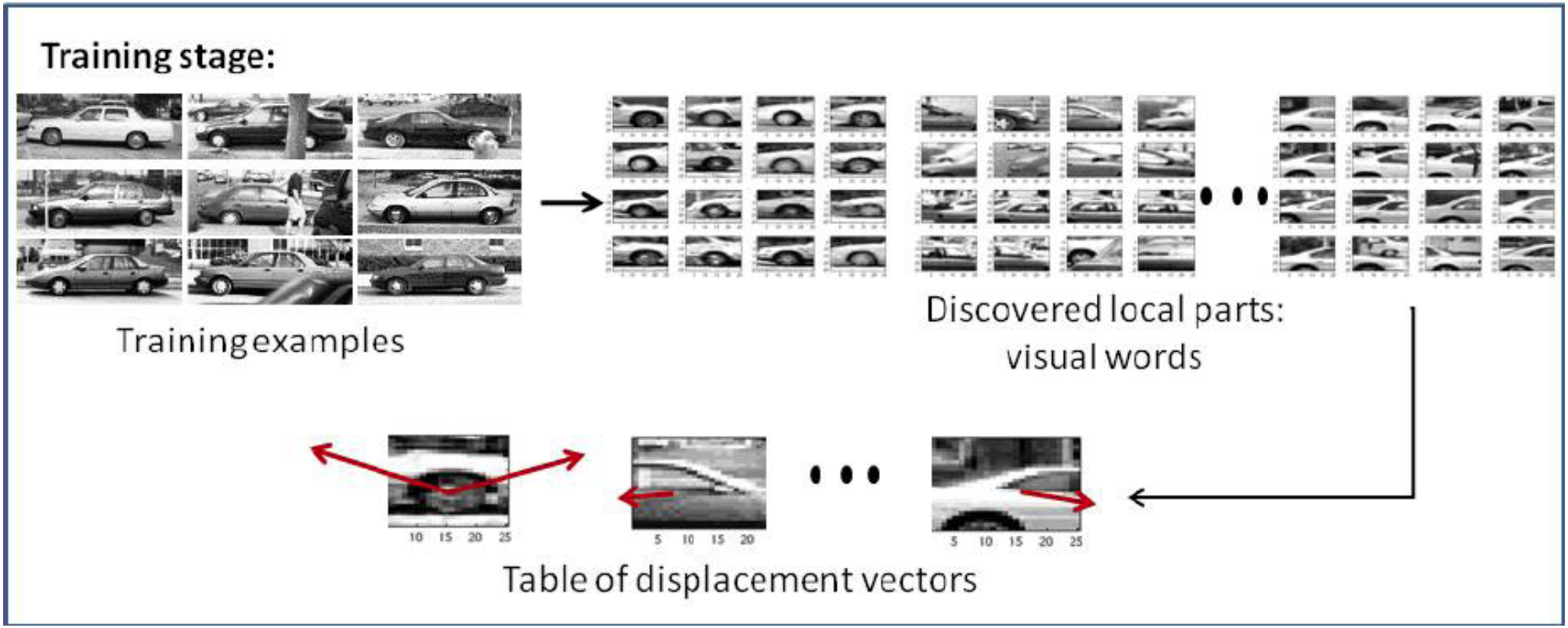
1. Build vocabulary of patches around extracted interest points using clustering
2. Map the patch around each interest point to closest word
3. For each word, store all positions it was found, relative to object center







Training


Training stage:
Training examples
Discovered local parts:
visual words
Table of displacement vectors

Implicit shape models: Testing

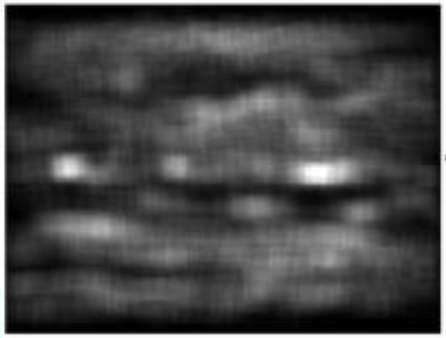
1. Given new test image, extract patches, match to vocabulary words
2. Cast votes for possible positions of object center
3. Search for maxima in voting space
4. (Extract weighted segmentation mask based on stored masks for the codebook occurrences)

What is the dimension of the Hough space?

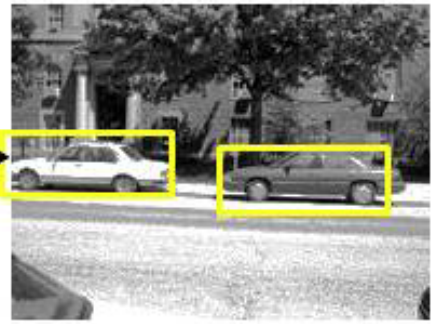
Detecting

Detection stage:

Novel image



Voting (GHT)

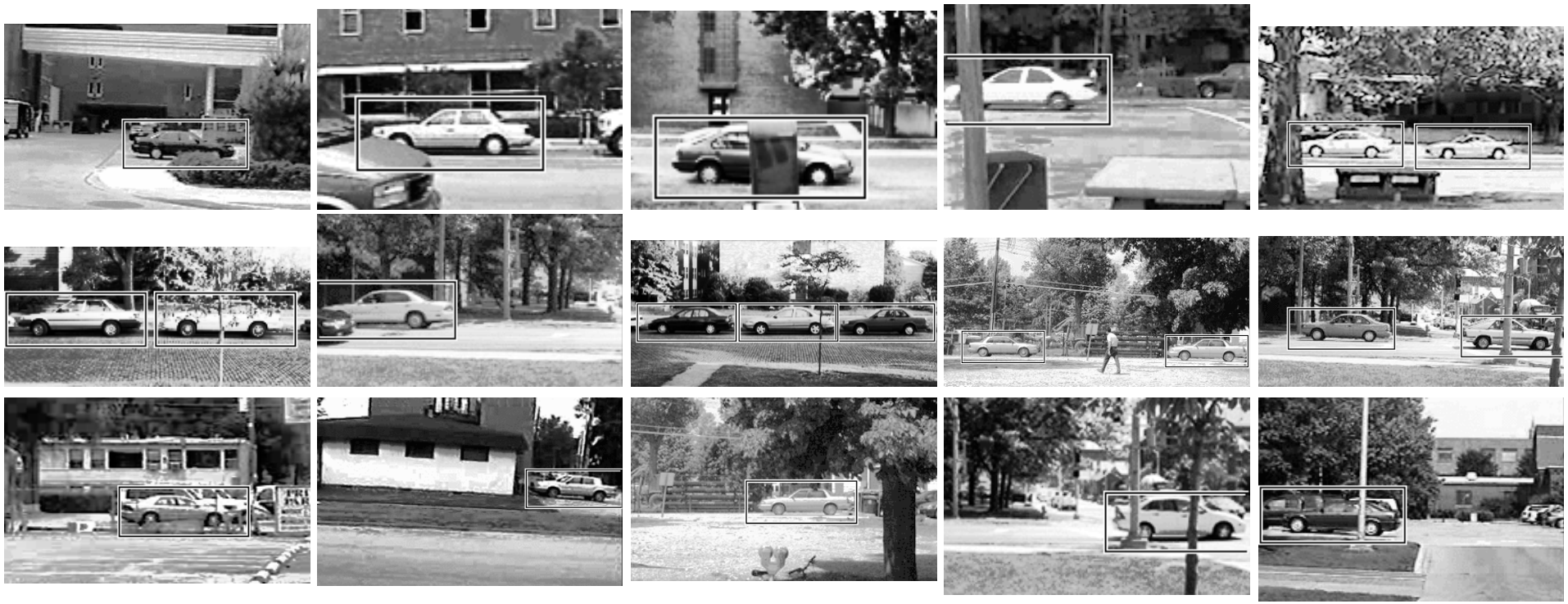


Predicted object locations

Detection Results

Qualitative Performance

- Recognizes different kinds of objects
- Robust to clutter, occlusion, noise, low contrast



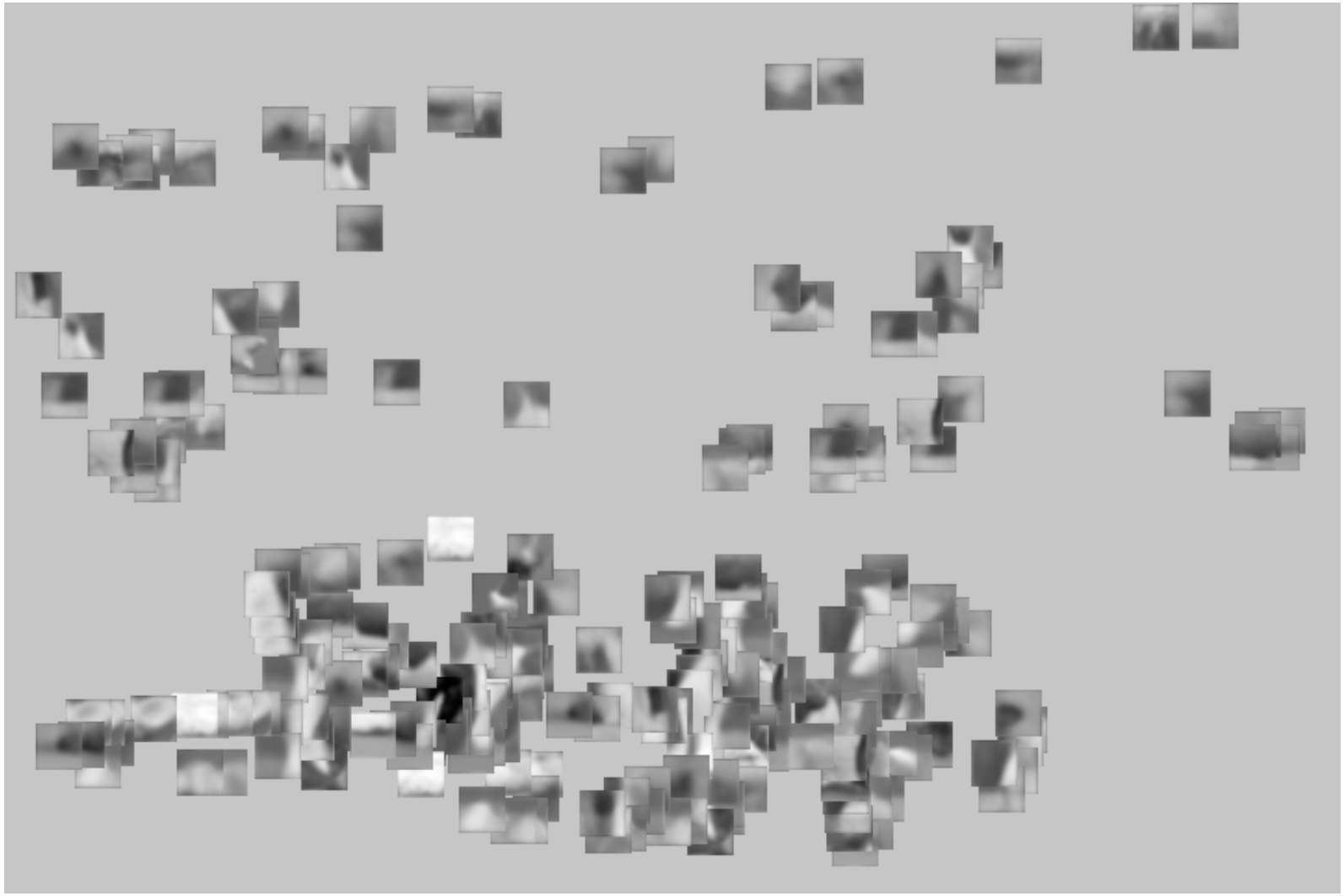
Example: Results on Cows

Original image

Example: Results on Cows

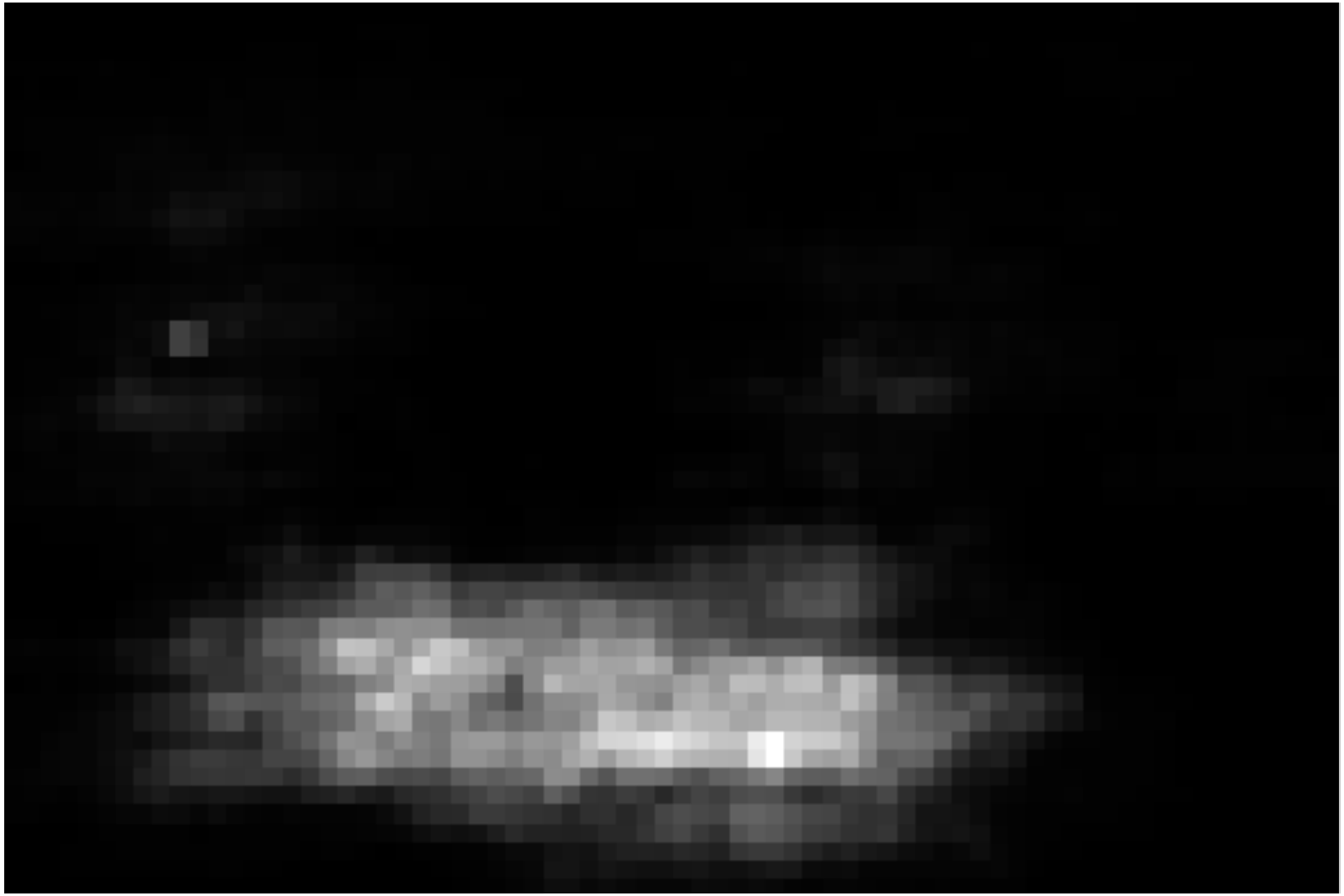
Interest points

Example: Results on Cows



Matched patches points

Example: Results on Cows



Matched **Votes**

Example: Results on Cows

1st hypothesis

Example: Results on Cows

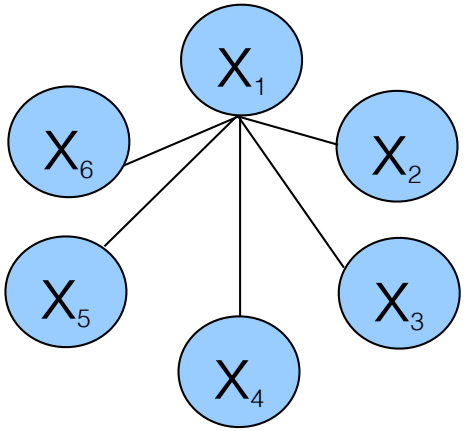
2nd hypothesis

Example: Results on Cows

3rd hypothesis

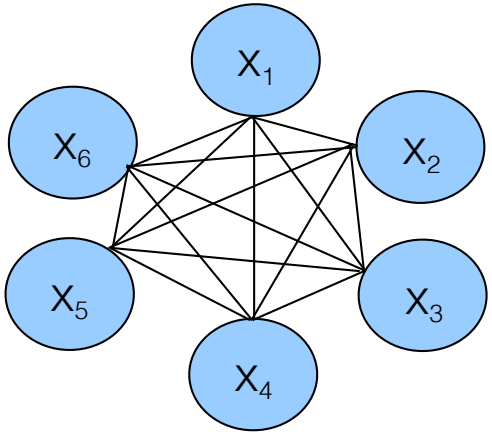
Shape representation in part-based models

"Star" shape model



- **e.g. implicit shape model**
- **Parts mutually independent**

Fully connected constellation model



- **e.g. Constellation Model**
- **Parts fully connected**

N image features, P parts in the model

Slide credit: Rob Fergus

Probabilistic constellation model

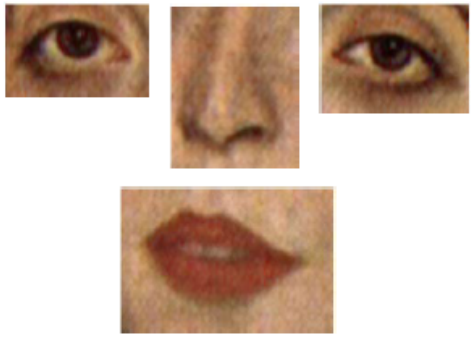
$$P(image \mid object) = P(appearance, shape \mid object)$$

Part descriptors

Part locations

Candidate parts

Constellation Model: Deformations



A

B

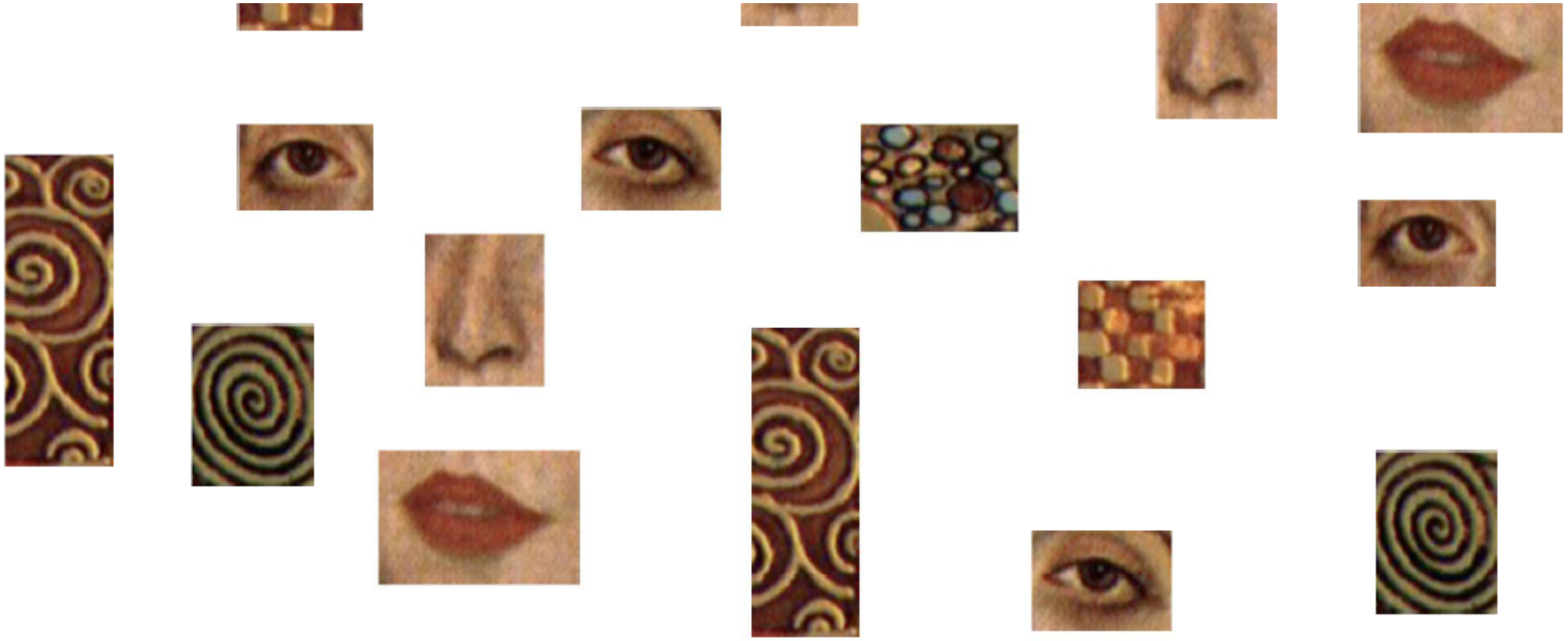
C

D

Constellation Model: Presence/Absence of Features

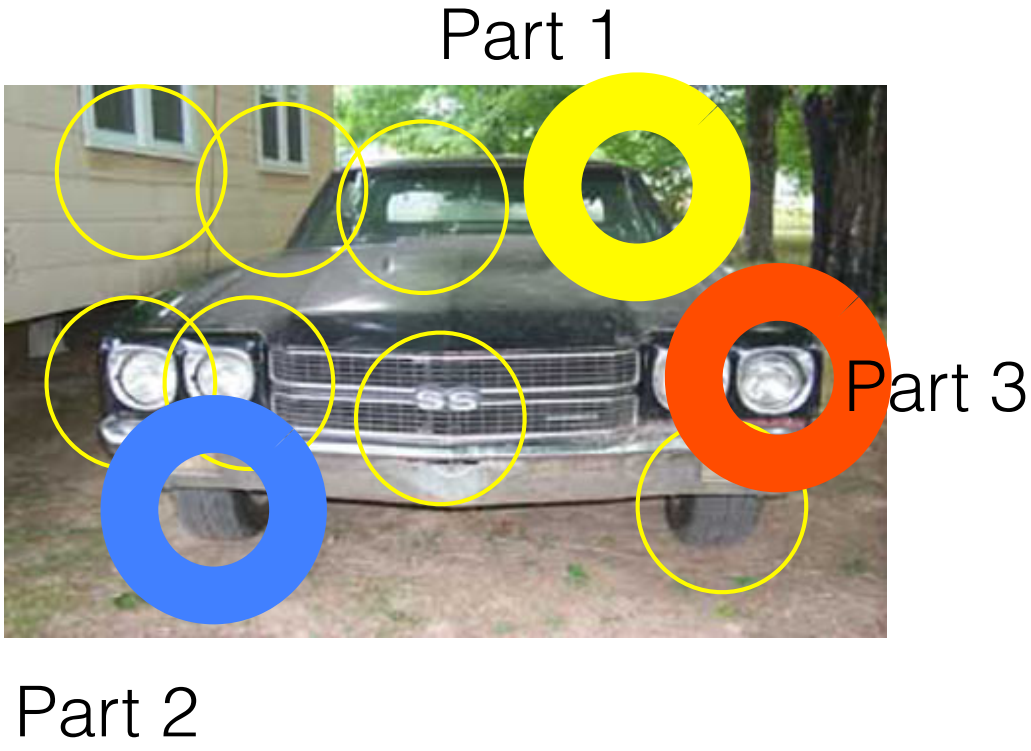
occlusion

Constellation Model: Clutter



Probabilistic constellation model

$$P(image \mid object) = P(appearance, shape \mid object)$$



Source: Lana Lazebnik

Probabilistic constellation model

$$P(image \mid object) = P(appearance, shape \mid object)$$

$$= \max_h P(appearance \mid h, object)\, p(shape \mid h, object)\, p(h \mid object)$$

h: assignment of features to parts

Part 1

Part 3

Part 2

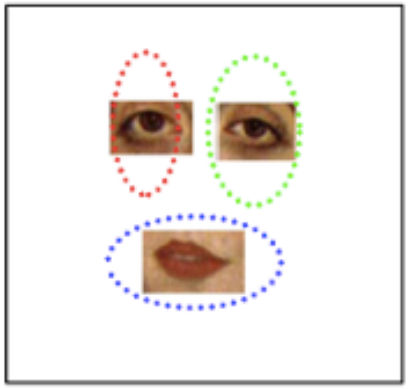
Source: Lana Lazebnik

Generative Model

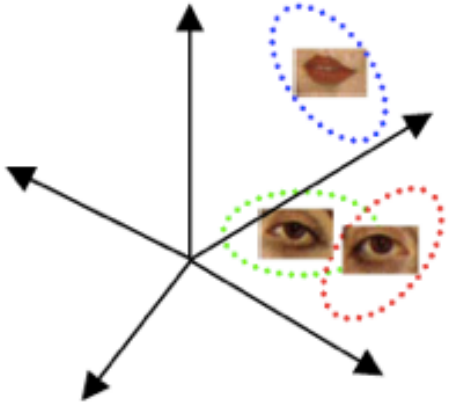
Foreground model

based on Burl, Weber et al. [ECCV '98, '00]

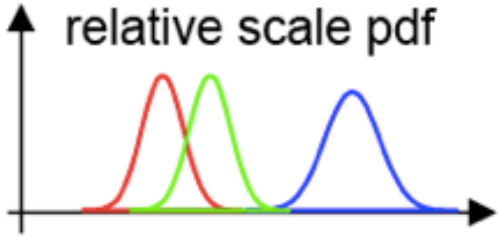
Gaussian shape pdf



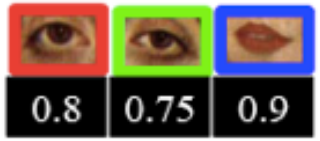
Gaussian part appearance pdf



Gaussian relative scale pdf



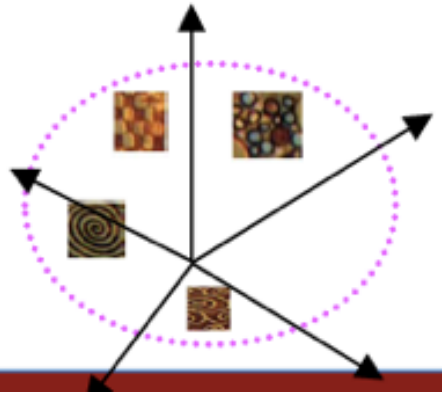
Prob. of detection



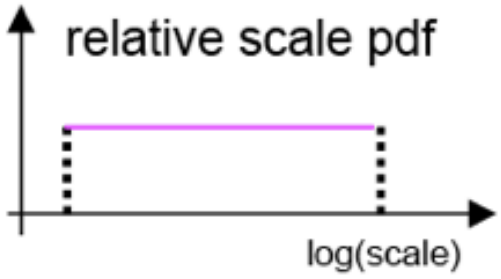
Clutter model

Uniform shape pdf

Gaussian background appearance pdf

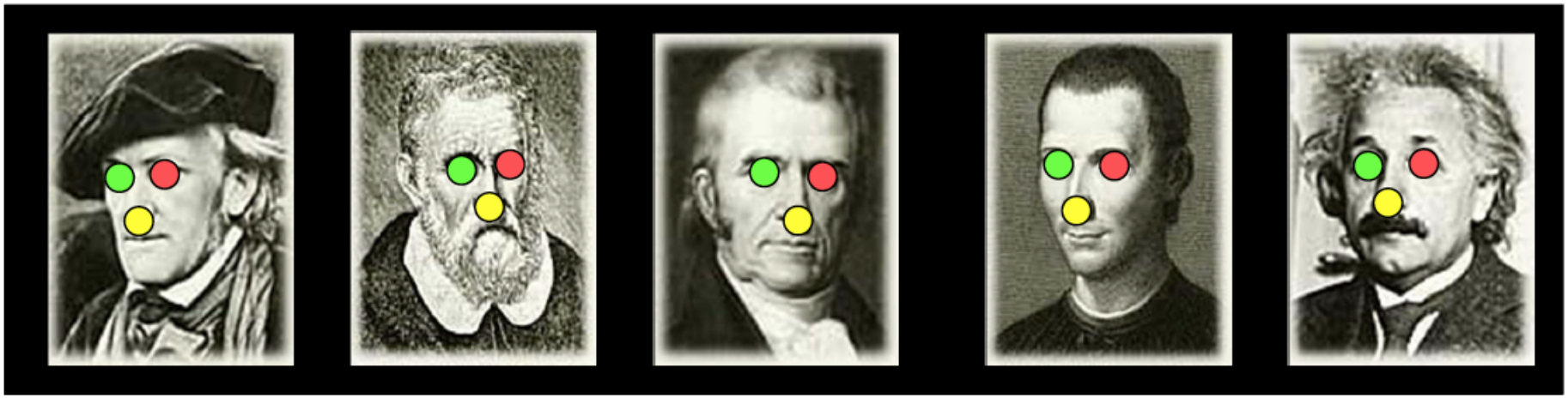


Uniform relative scale pdf

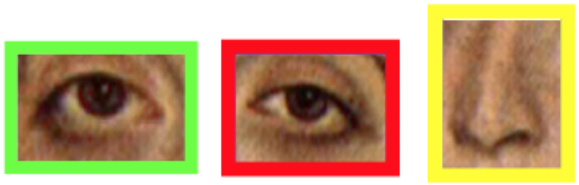


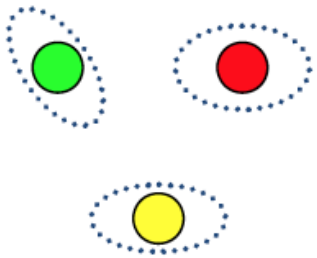
Poission pdf on # detections

Learning Models Manually



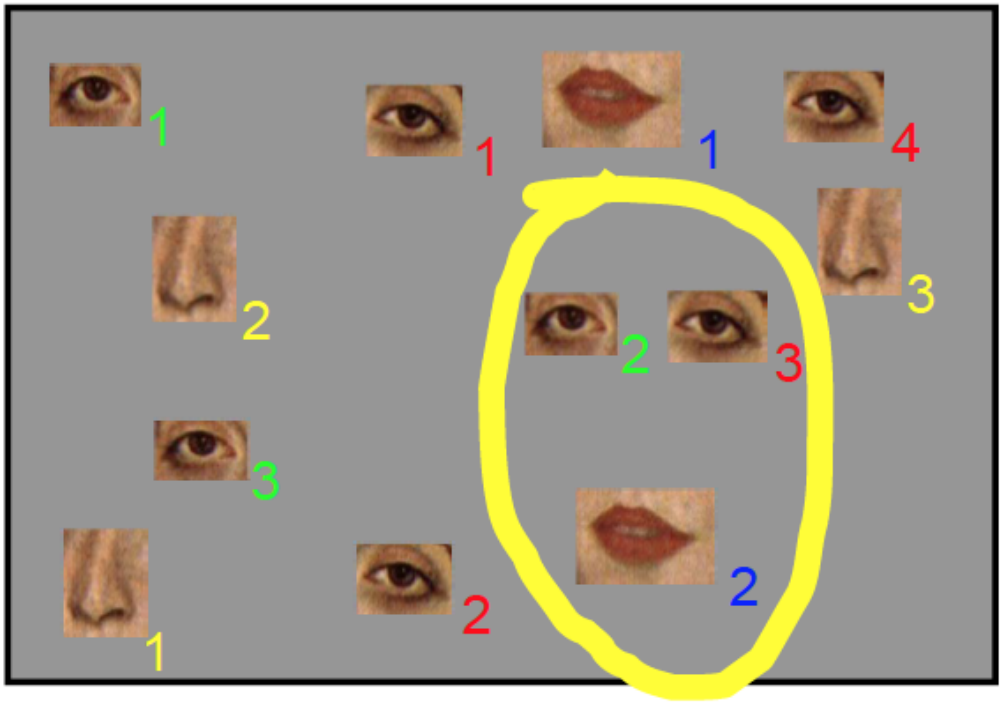
- Obtain set of training images
- Choose parts
- Label parts by hand, train detectors
- Learn model from labeled parts





Recognition

1. Run part detectors exhaustively over image



$$h = \begin{pmatrix} 0 \ldots N_1 \\ 0 \ldots N_2 \\ 0 \ldots N_3 \\ 0 \ldots N_4 \end{pmatrix} \qquad \text{e.g. } h = \begin{pmatrix} 2 \\ 3 \\ 0 \\ 2 \end{pmatrix}$$

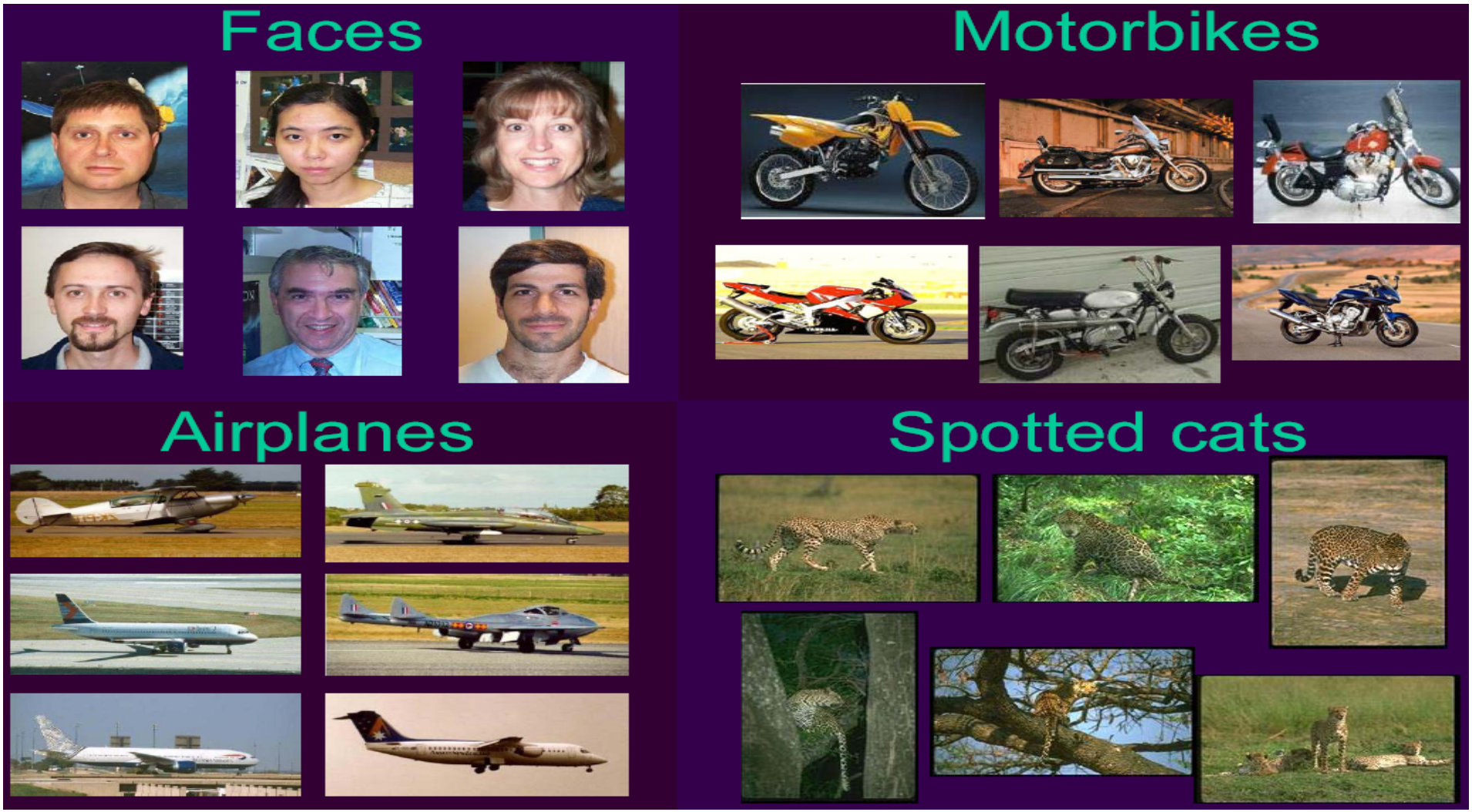
2. Try different combinations of detections in model
   - Allow detections to be missing (occlusion)

3. Pick hypothesis which maximizes:

$$\frac{p(Data \mid Object, Hyp)}{p(Data \mid Clutter, Hyp)}$$

4. If ratio is above threshold then, instance detected

Example results from constellation model: data from four categories



Slide from Li Fei-Fei http://www.vision.caltech.edu/feifeili/Resume.htm

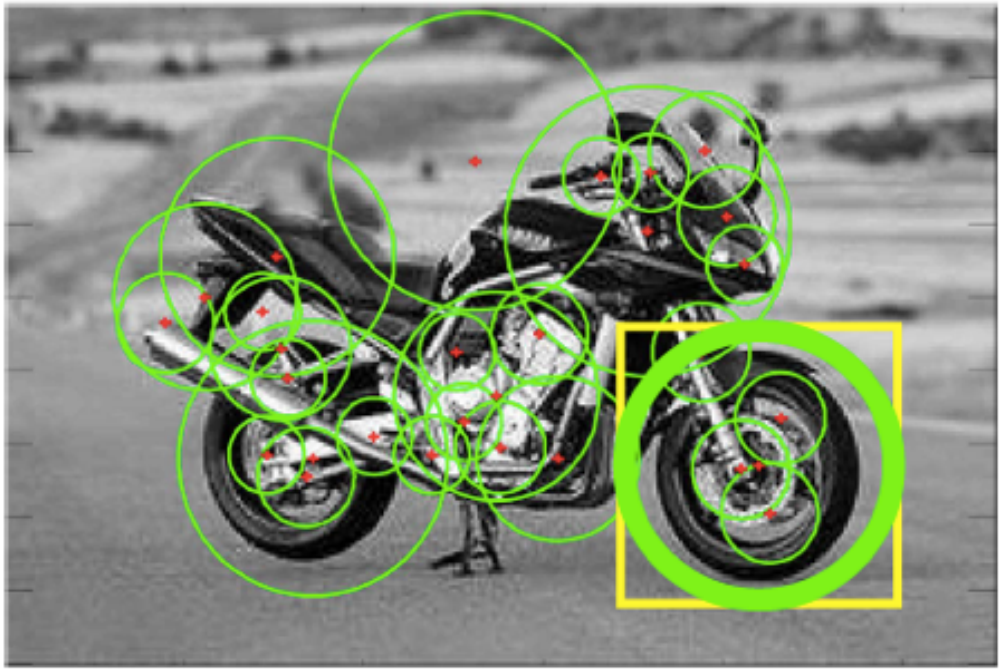
Weakly Supervised Class Learning

Train class without segmentation

Training images have the object of interest (once)

but do not know where or the scale

Semi-supervised Training (Fergus et al '03)



- Find regions within image
- Use salient region operator (Kadir & Brady 01)

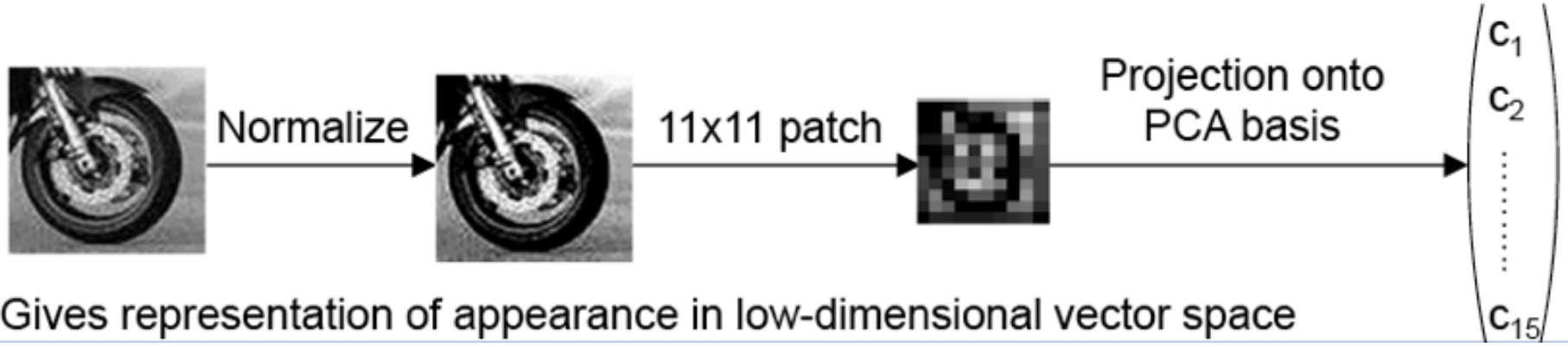
Location

(x,y) coords. of region centre

Scale

Radius of region (pixels)

Appearance



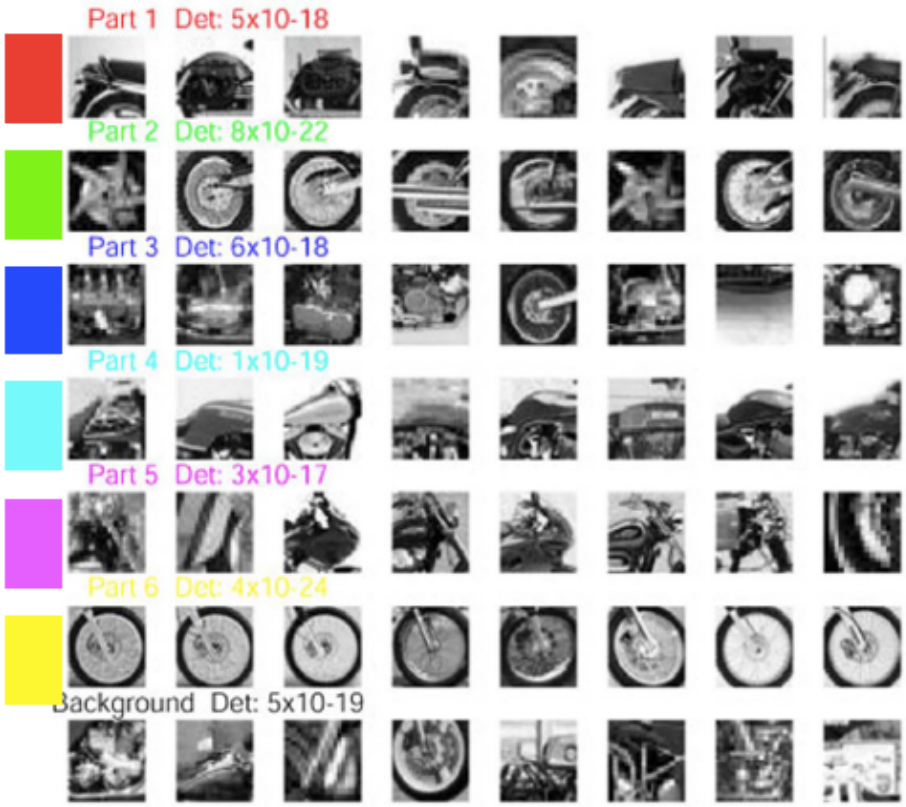
Gives representation of appearance in low-dimensional vector space

Motorcycle Example

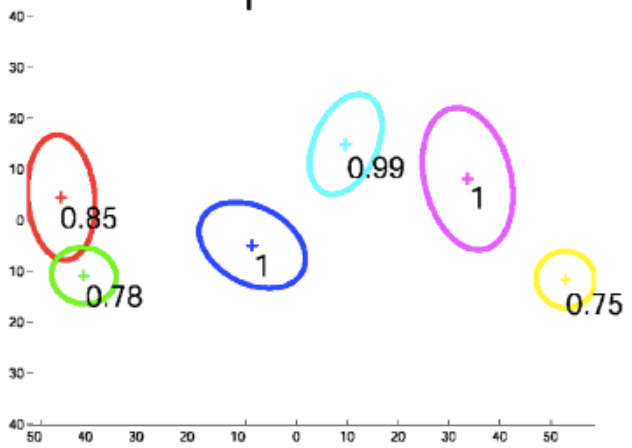
•Kadir & Brady saliency region detector

Motorbikes

Samples from appearance model

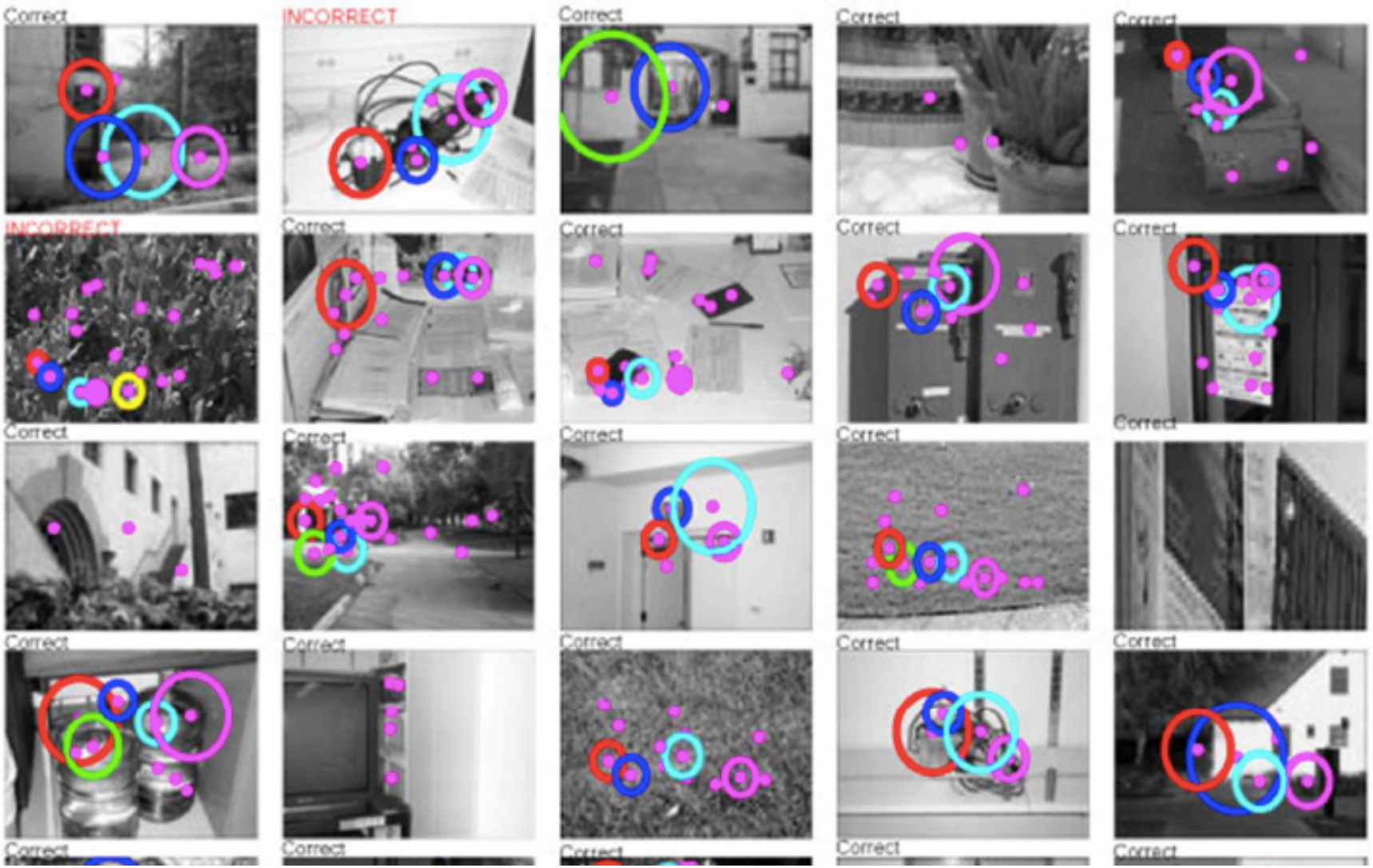


Shape model

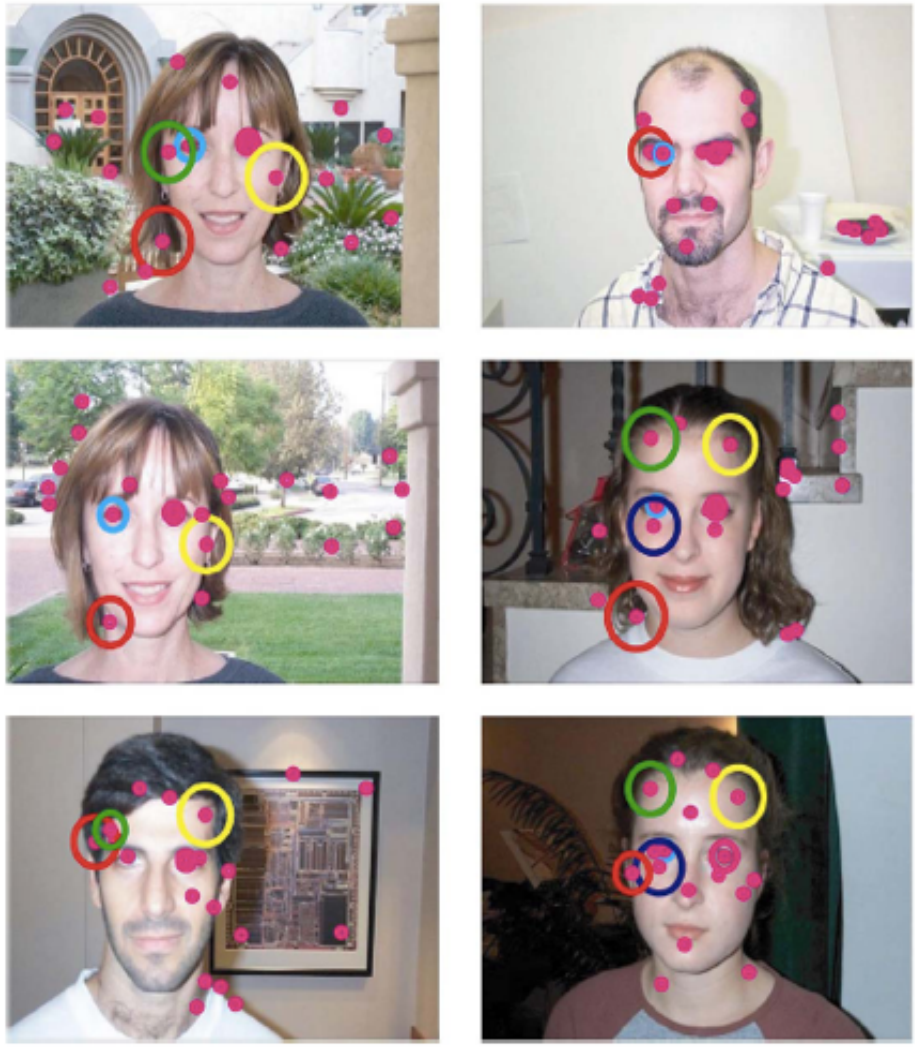


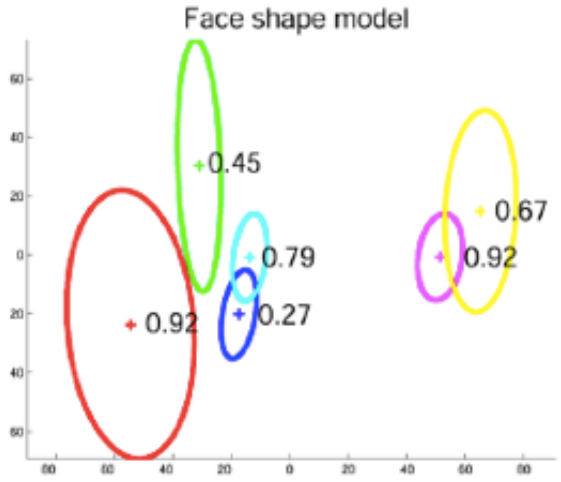
Test images: size of circles indicates score of hypothesis

Motorbike Model on Background Images

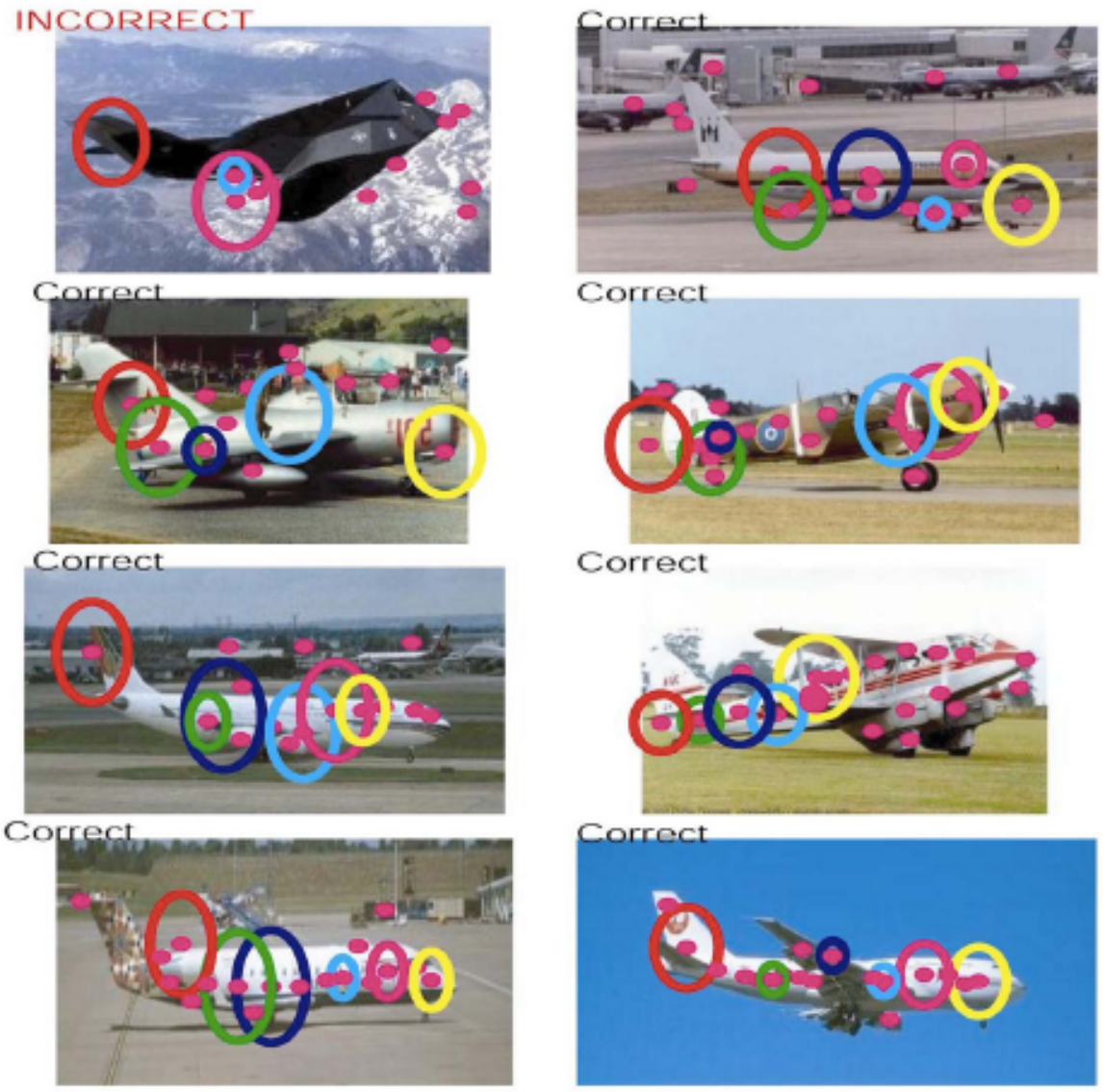


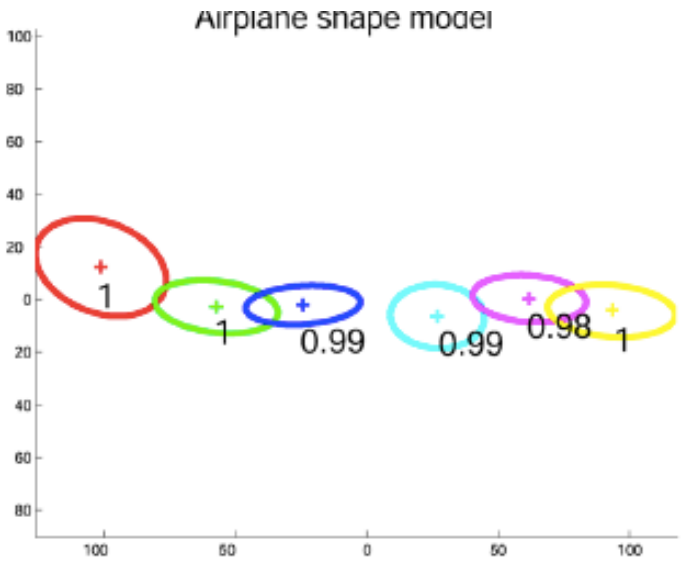
Frontal Faces

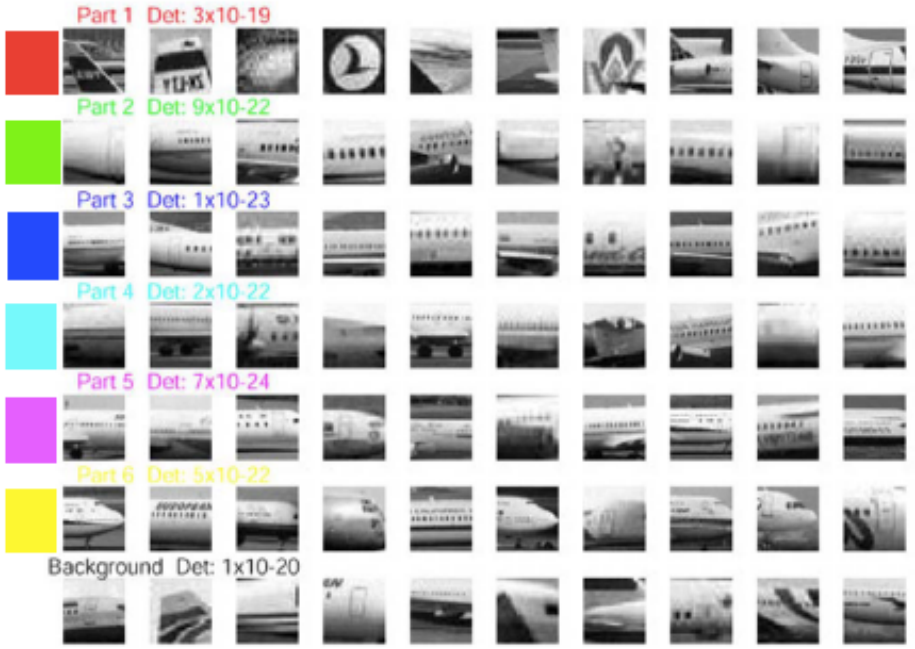




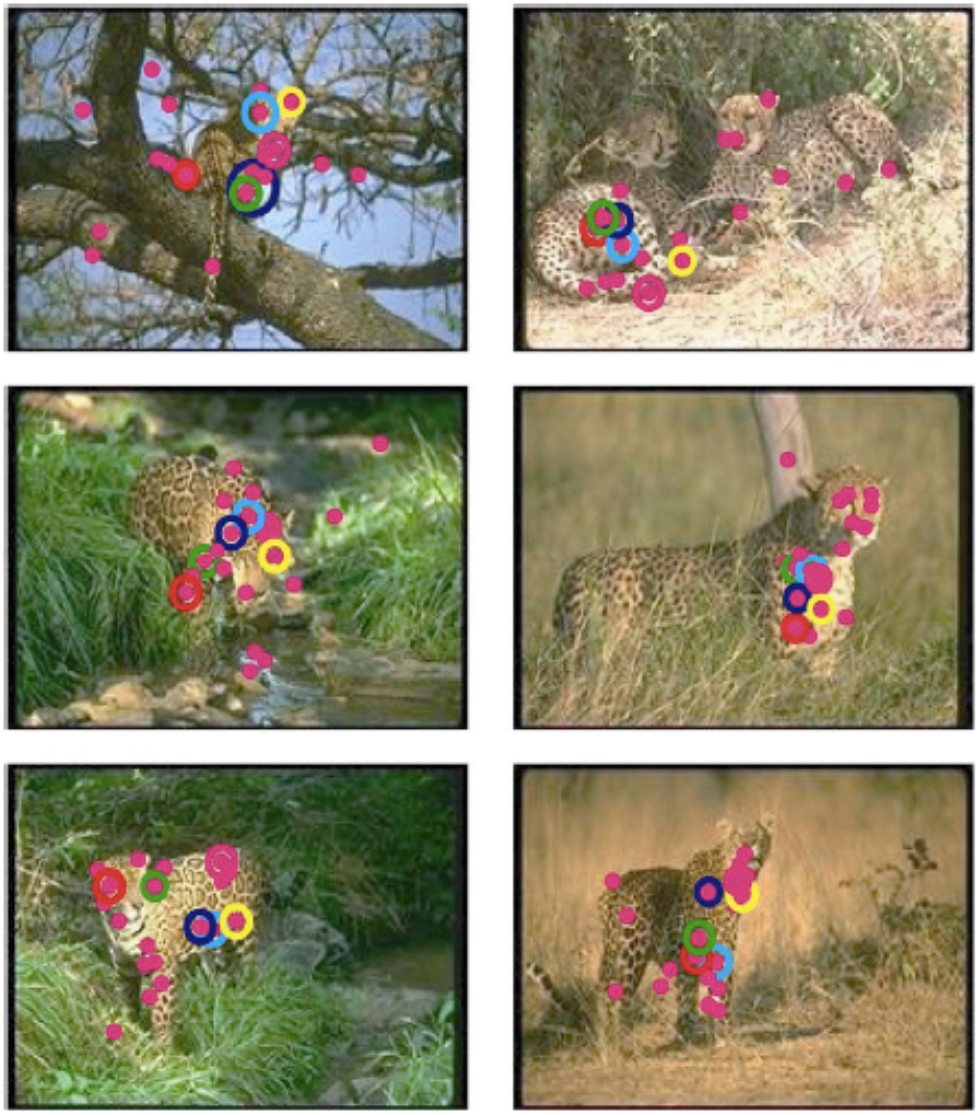
Airplanes

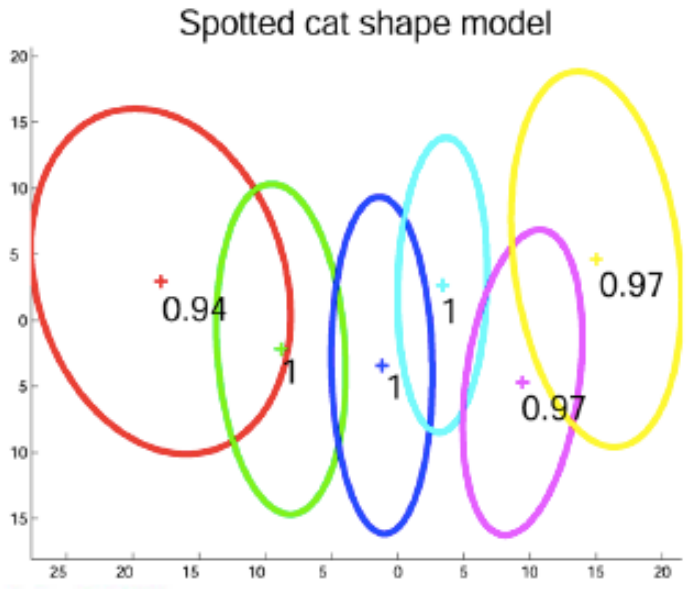

INCORRECT
Correct
Correct
Correct
Correct
Correct
Correct
Correct

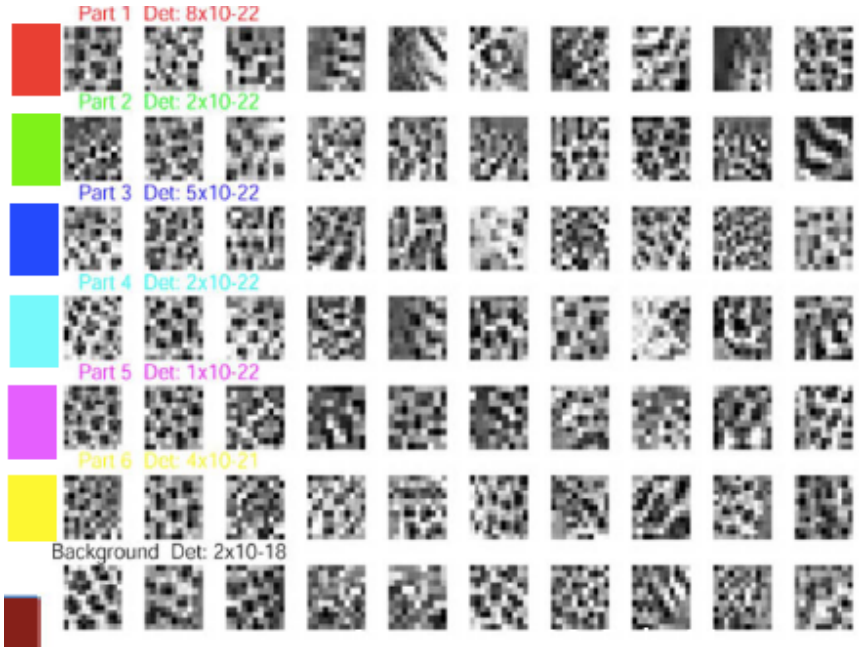

Airplane shape model
100
80
60
40
20
0
20
40
60
80
100
50
0
50
100
1
1
0.99
0.99
0.98
1


Part 1 Det: 3x10-19
Part 2 Det: 9x10-22
Part 3 Det: 1x10-23
Part 4 Det: 2x10-22
Part 5 Det: 7x10-24
Part 6 Det: 5x10-22
Background Det: 1x10-20

Spotted Cats







Comparison



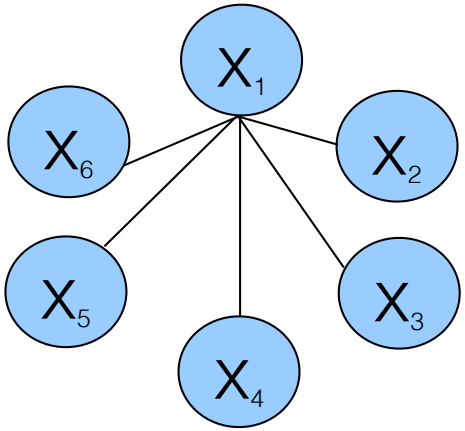



class	bag of features Zhang et al. (2005)	bag of features Willamowski et al. (2004)	Part-based model Fergus et al. (2003)
airplanes	**98.8**	97.1	90.2
cars (rear)	98.3	**98.6**	90.3
cars (side)	**95.0**	87.3	88.5
faces	**100**	99.3	96.4
motorbikes	**98.5**	98.0	92.5
spotted cats	**97.0**	—	90.0

Source: Lana La

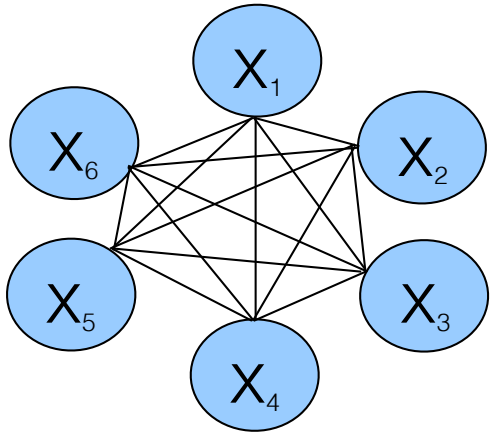
Shape representation in part-based models

"Star" shape model



- **e.g. implicit shape model**
- **Parts mutually independent**
- **Recognition complexity: O(NP)**
- **Method: Gen. Hough Transform**

Fully connected constellation model



- **e.g. Constellation Model**
- **Parts fully connected**
- **Recognition complexity: O(N^P)**
- **Method: Exhaustive search**

N image features, P parts in the model

Slide credit: Rob Fergus

Summary:
part-based and local feature models for generic object recognition

Histograms of visual words to capture global or local layout in the bag-of-words framework

- Powerful in practice for image recognition

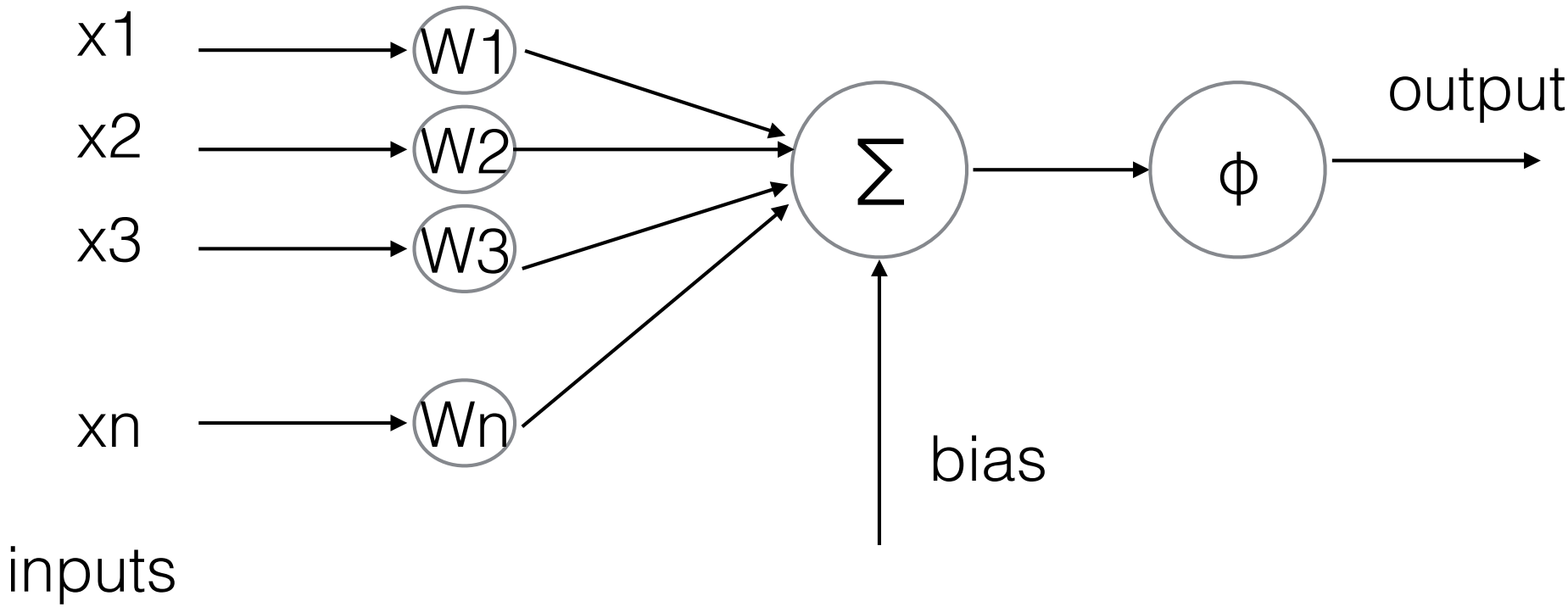
Part-based models encode category's part appearance together with 2d layout and allow detection within cluttered image

- **"implicit shape model"**: shape based on layout of all parts relative to a reference part; Generalized Hough for detection
- **"constellation model"**: explicitly model mutual spatial layout between all pairs of parts; exhaustive search for best fit of features to parts

Deep Learning

Convolutional Neural Nets (CNN)

Neuron



$$o = \phi(\sum(x_i.W_i) + b)$$

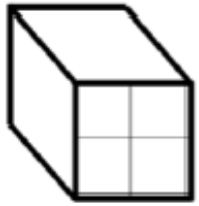
ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton

From Filtering to CNNs

Input “image”

Filter

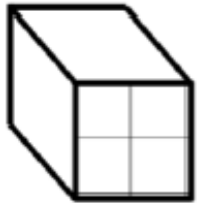


From Filtering to CNNs

If the filter is [-1 1]

Input "image"

Filter

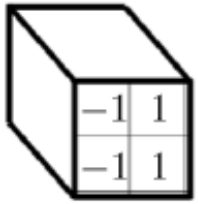


From Filtering to CNNs

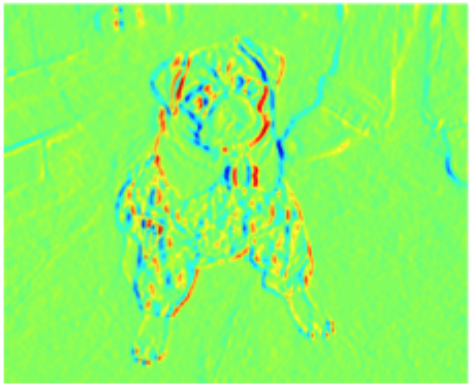
If the filter is [-1 1] Vertical Edges

Input "image"

Filter



Output map

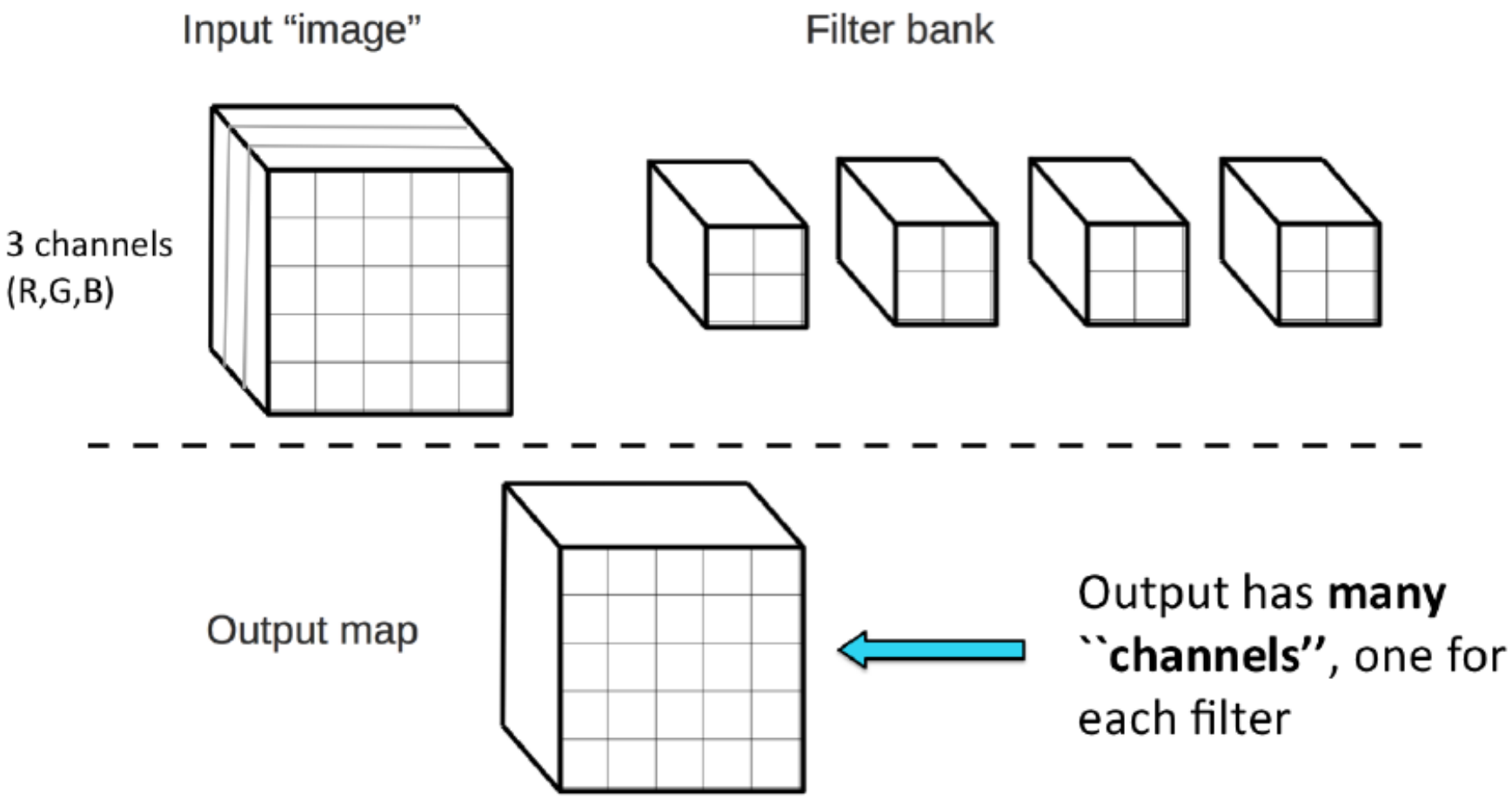


From Filtering to CNNs

How about having lots of filters?
- vertical edges,
- horizontal edges,
- corners,
- dots,
- etc

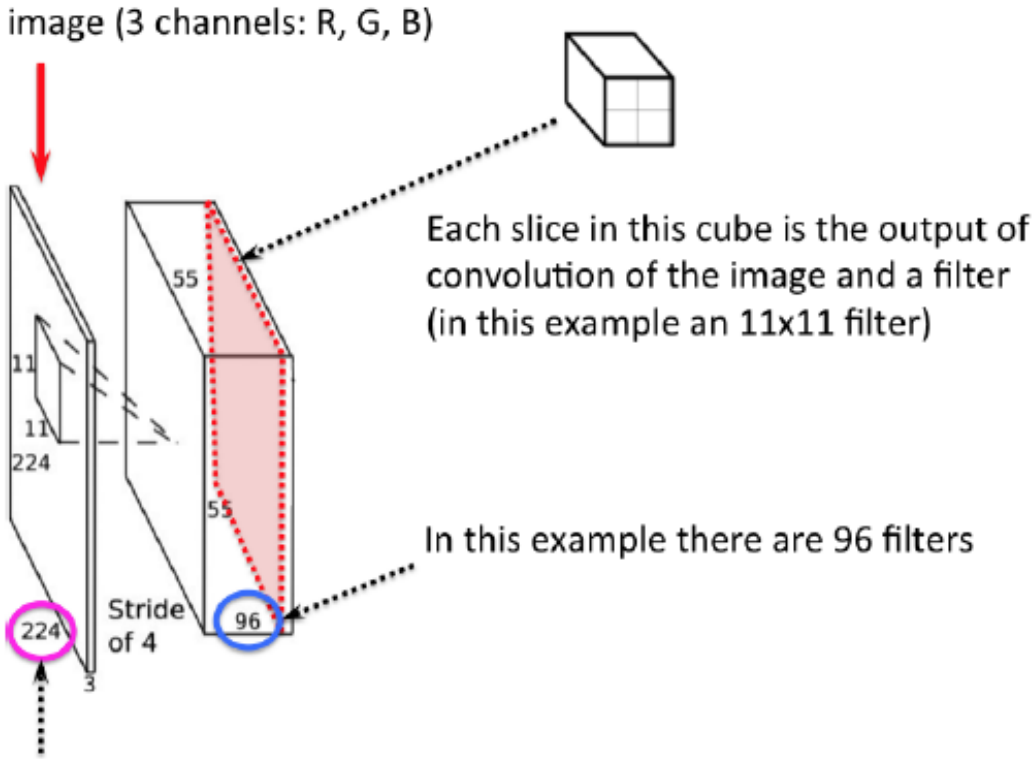
A Filter Bank!

From Filtering to CNNs



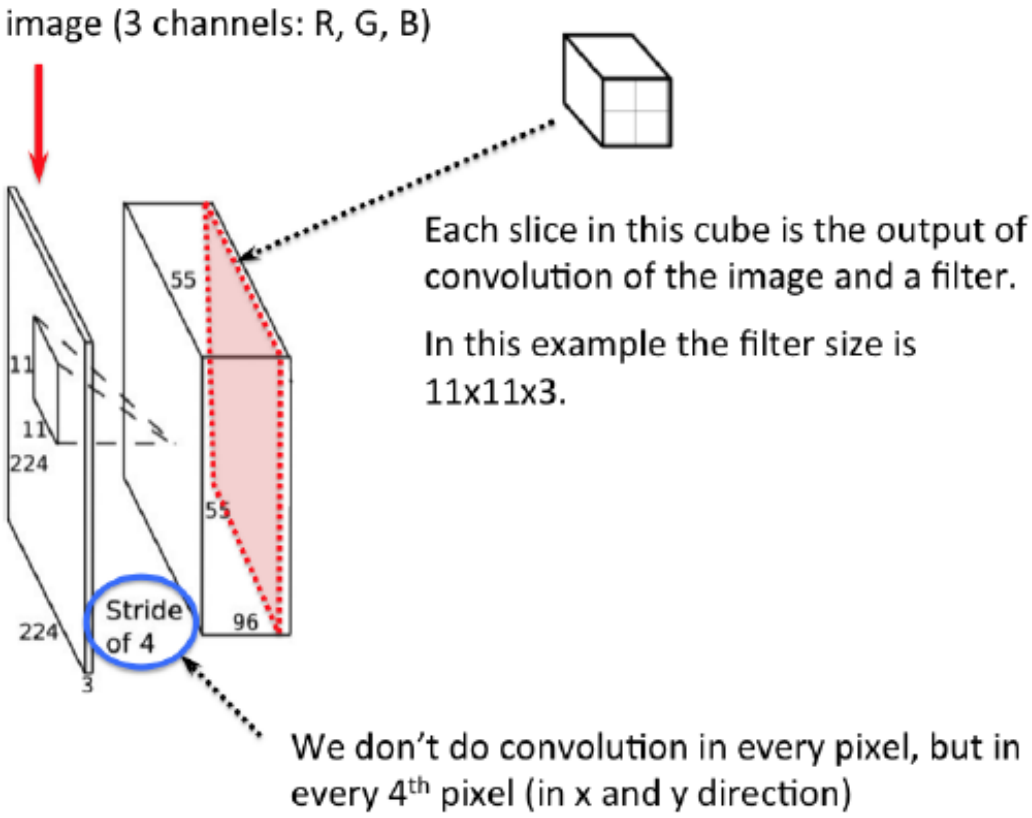
CNN

Applying a filter bank to an image yields a cube-like output: each slice is the output of a convolution with a filter

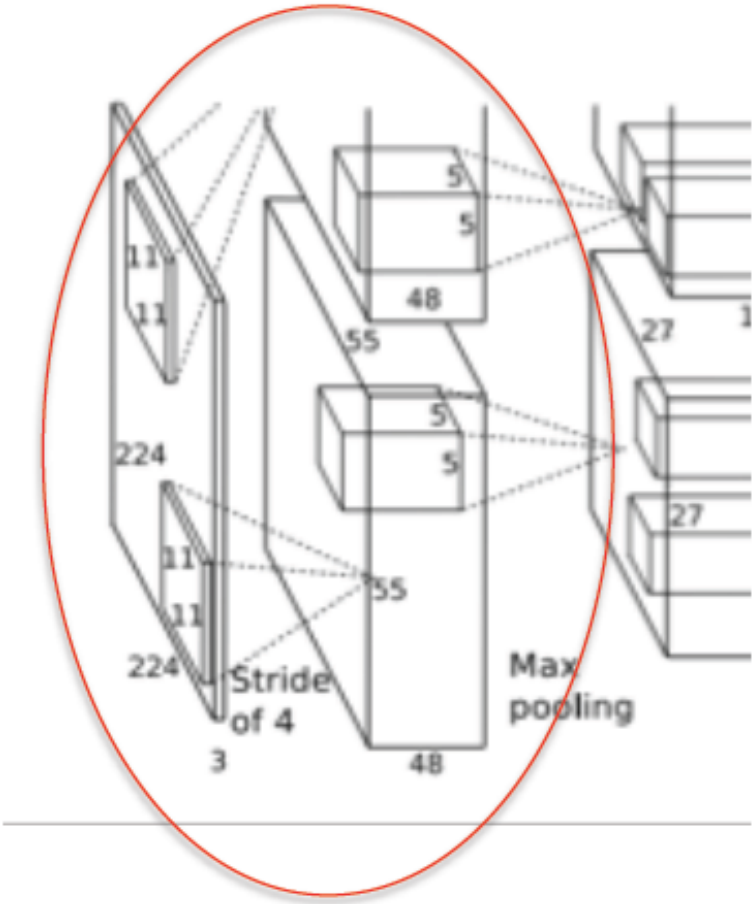


CNN

Applying a filter bank to an image yields a cube-like output: each slice is the output of a convolution with a filter

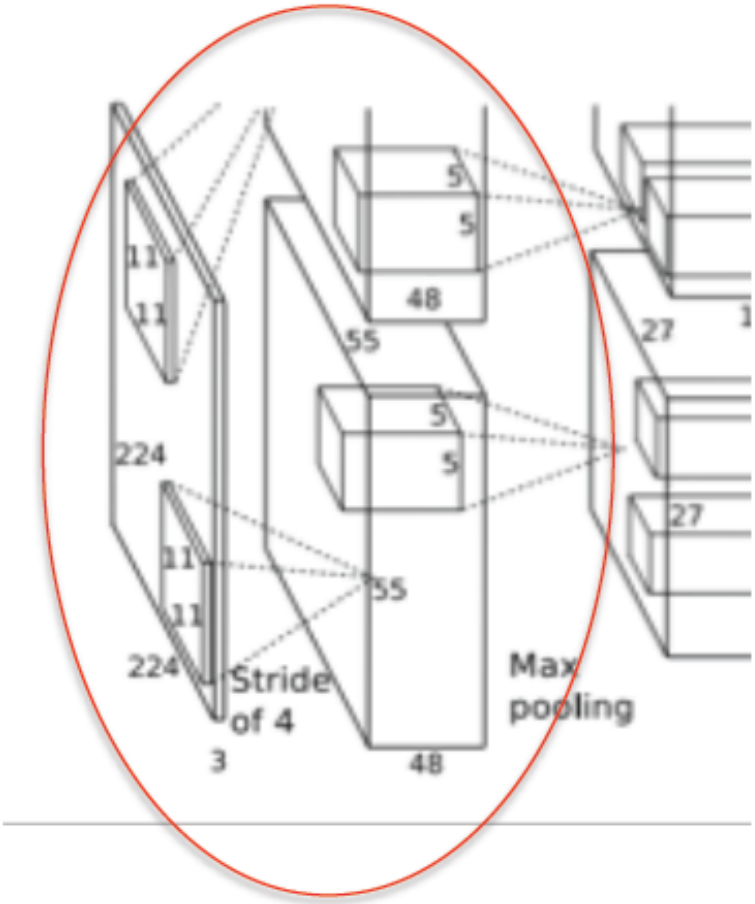


Convolutional Layer 1



- Images: 227x227x3
- F (receptive field size): 11
- S (stride) = 4
- Conv layer output: 55x55x96

Convolutional Layer 1



- 55*55*96 = 290,400 neurons
- each has 11*11*3 = 363 weights and 1 bias
- 290400 * 364 = 105,705,600 paramaters on the first layer of the AlexNet alone!

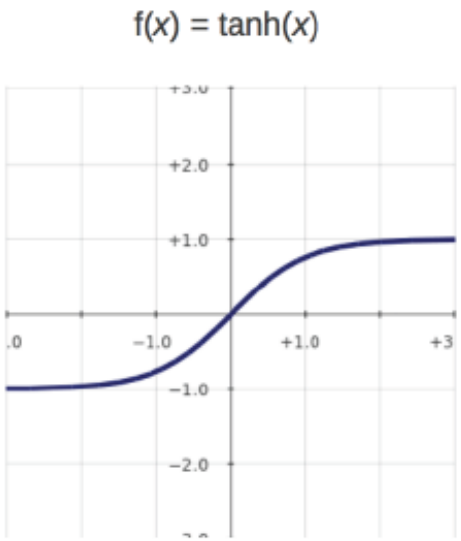
ReLU: Rectifier Linear Unit

RELU Nonlinearity

- Standard way to model a neuron

$f(x) = tanh(x)$ or $f(x) = (1 + e^{-x})^{-1}$

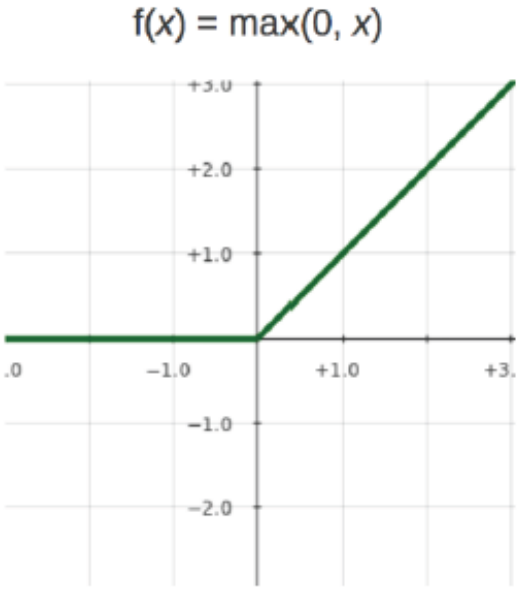
Very slow to train



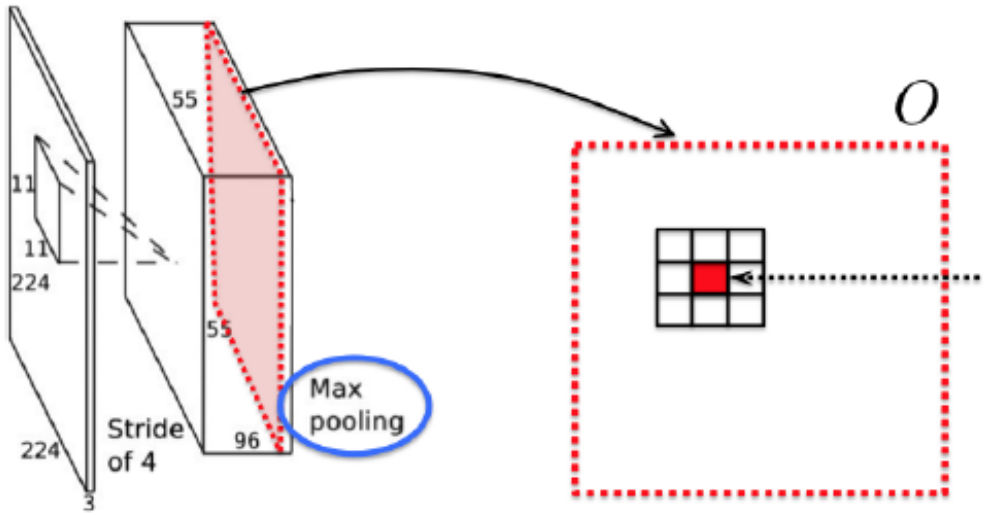
- Non-saturating nonlinearity (RELU)

$f(x) = max(0, x)$

Quick to train



CNN: Maxpooling

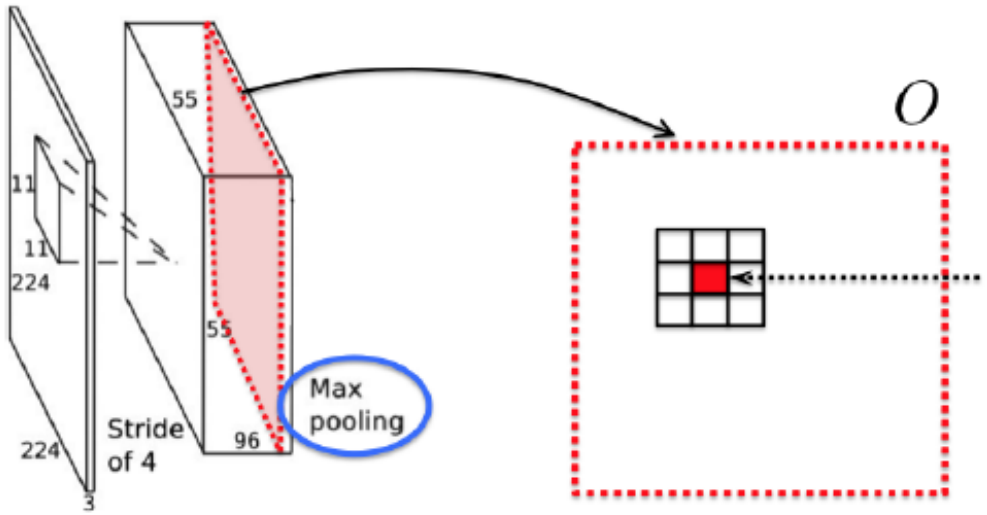


$$O(i,j) = \max_{\substack{k \in \{i-1,i,i+1\} \\ l \in \{j-1,j,j+1\}}} O(k,l)$$

Take each slice in the output cube, and in each pixel compute a max over a small patch around it. This is called **max pooling**.

WHY?

CNN: Maxpooling



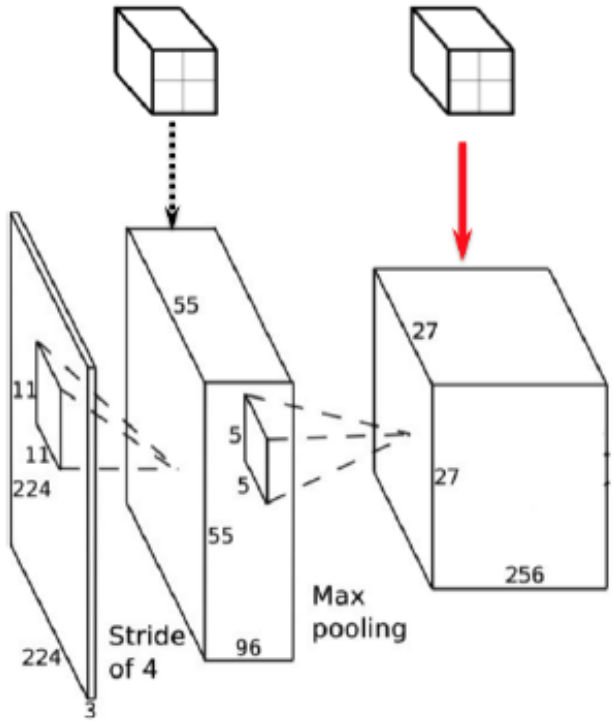
$$O(i,j) = \max_{\substack{k \in \{i-1,i,i+1\} \\ l \in \{j-1,j,j+1\}}} O(k,l)$$

Take each slice in the output cube, and in each pixel compute a max over a small patch around it. This is called **max pooling**.

To bring invariance to small shifts

CNN: More Layers

More convolutions, but now the input is the output of the previous layer.



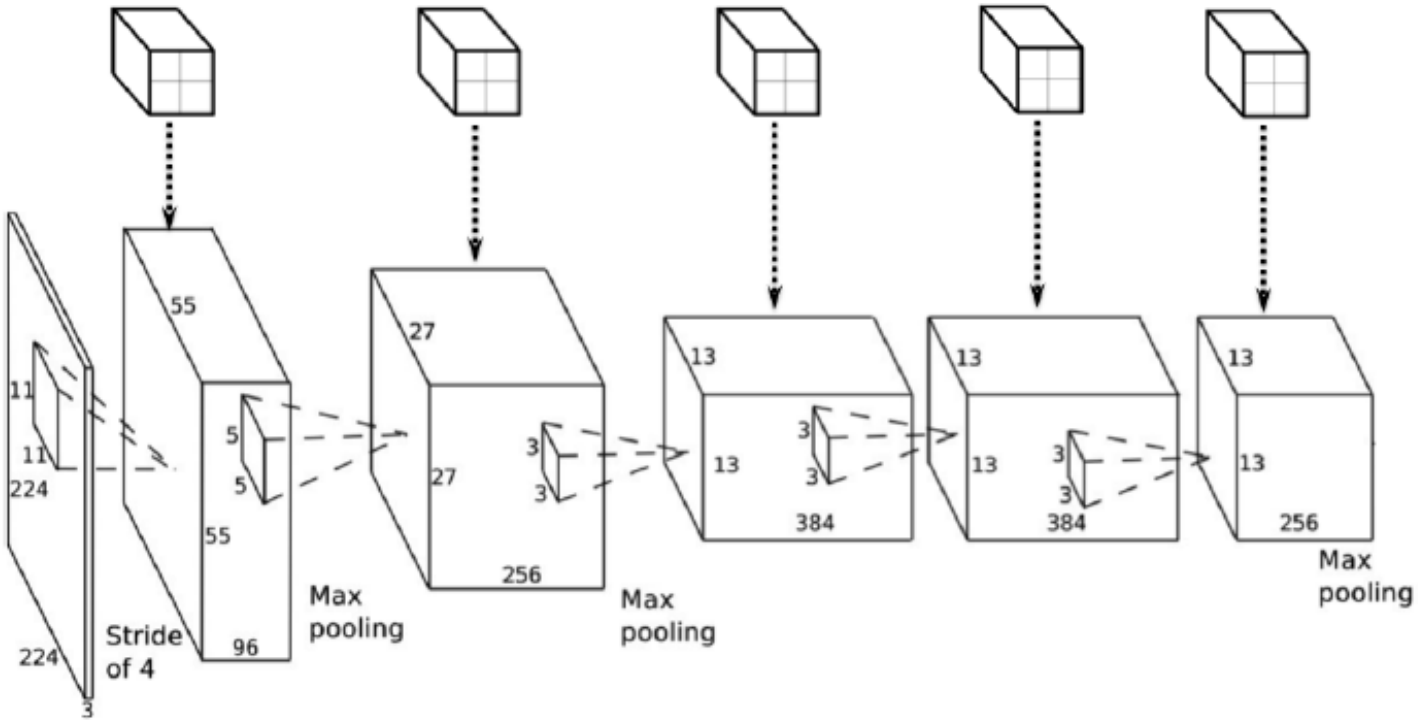
Add one more layer of filters

These filters are convolved with the output of the previous layer. The results of each convolution is again a slice in the cube on the right.

What is the dimension of each of these filters?

CNN: More Layers

More convolutions, but now the input is the output of the previous layer.

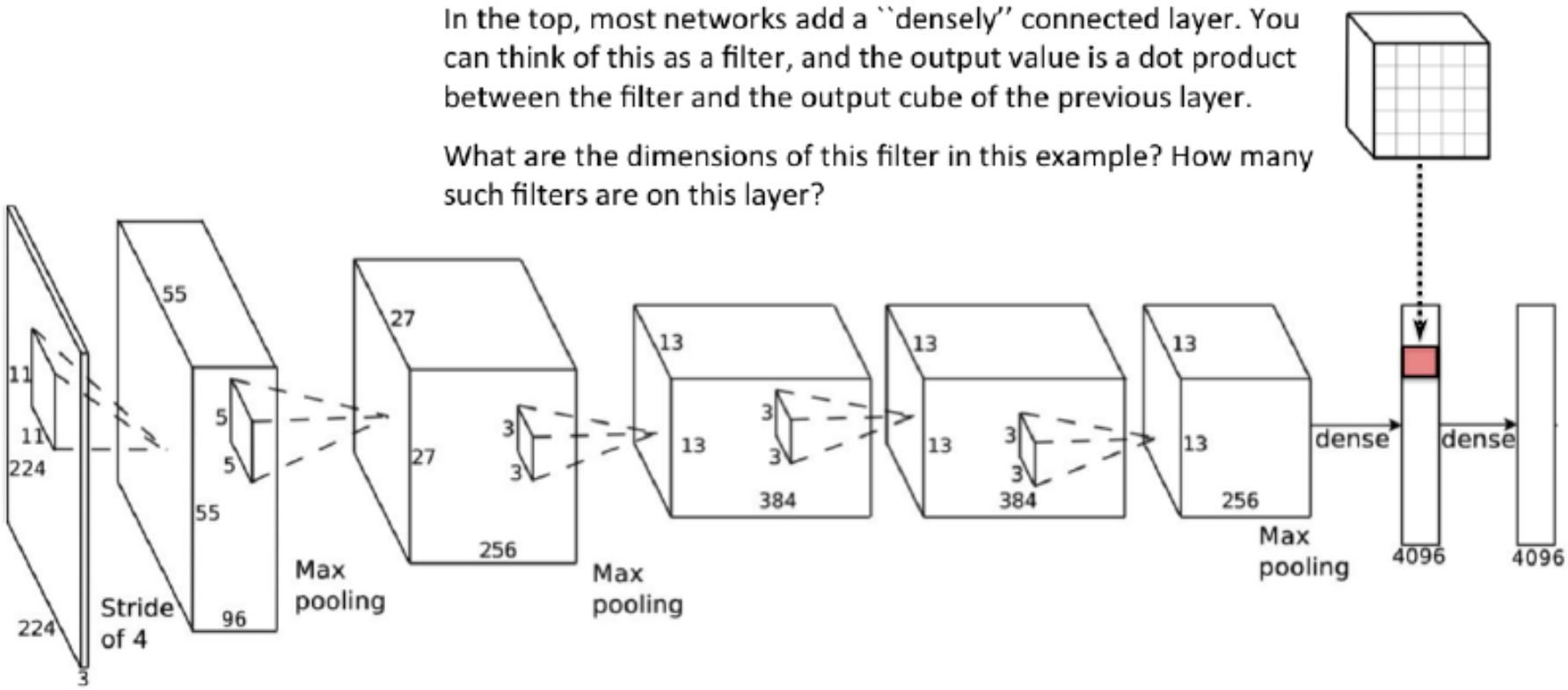


Do it recursively

Have multiple ``layers''

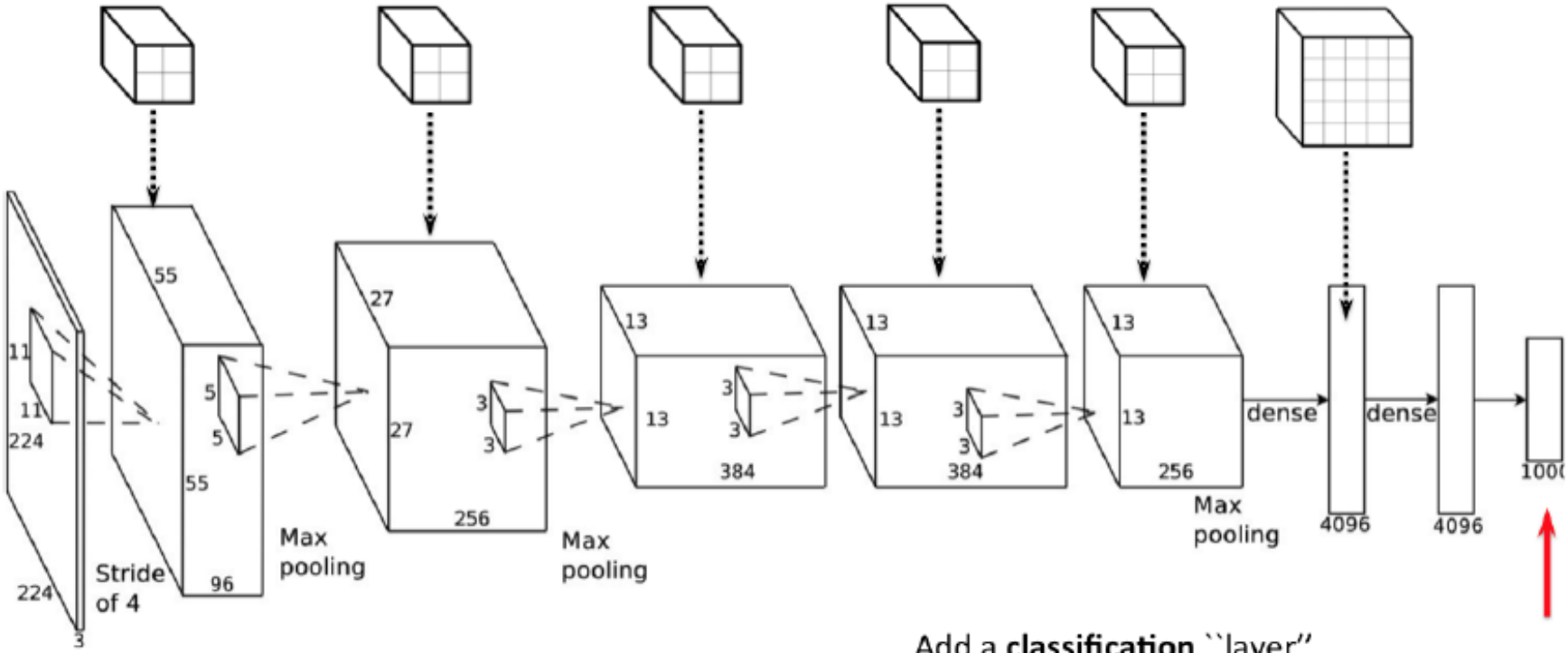
Fully Connected Layers

In the end, add two fully connected layers. These layers do not perform convolution. Instead, they just do a dot-product between the "filter" and the output of the previous layer.

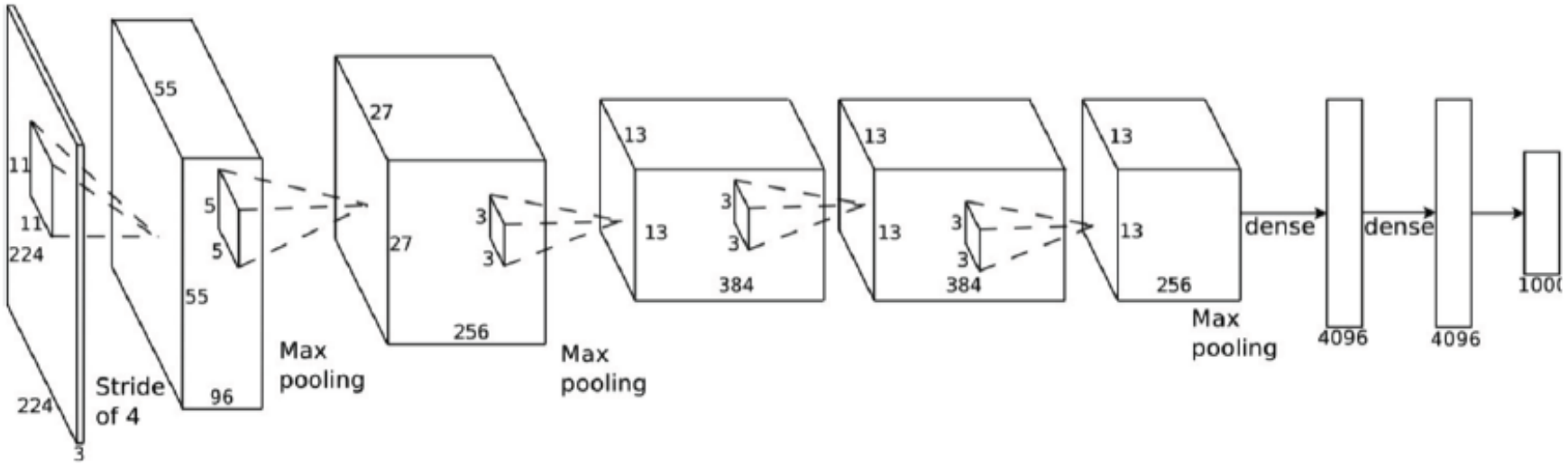


Last Layer: Classifier Layer

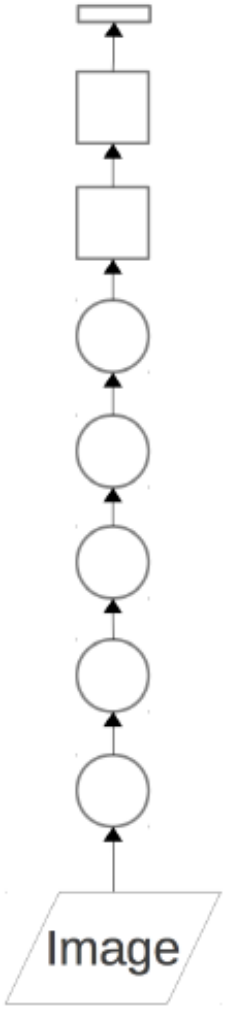
The final layer is a classification layer. Each dimension of this vector tells us the probability of the input image being a certain class.



How Many Parameters?



How Many Parameters?



- Trained with stochastic gradient descent on two NVIDIA GPUs for about a week
- 650,000 neurons
- 60,000,000 parameters
- 630,000,000 connections
- **Final feature layer:** 4096-dimensional

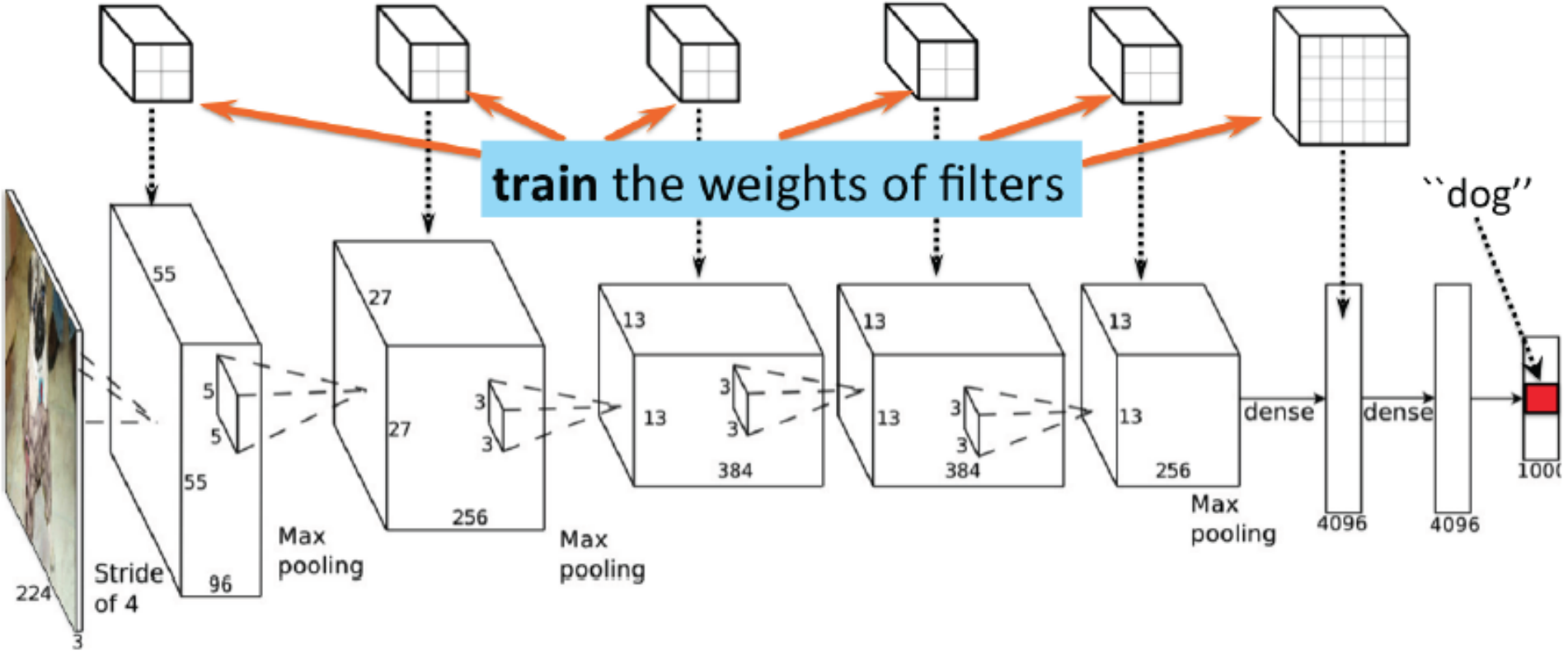
Convolutional layer: convolves its input with a bank of 3D filters, then applies point-wise non-linearity

Fully-connected layer: applies linear filters to its input, then applies point-wise non-linearity

Figure: From: `http://www.image-net.org/challenges/LSVRC/2012/supervision.pdf`

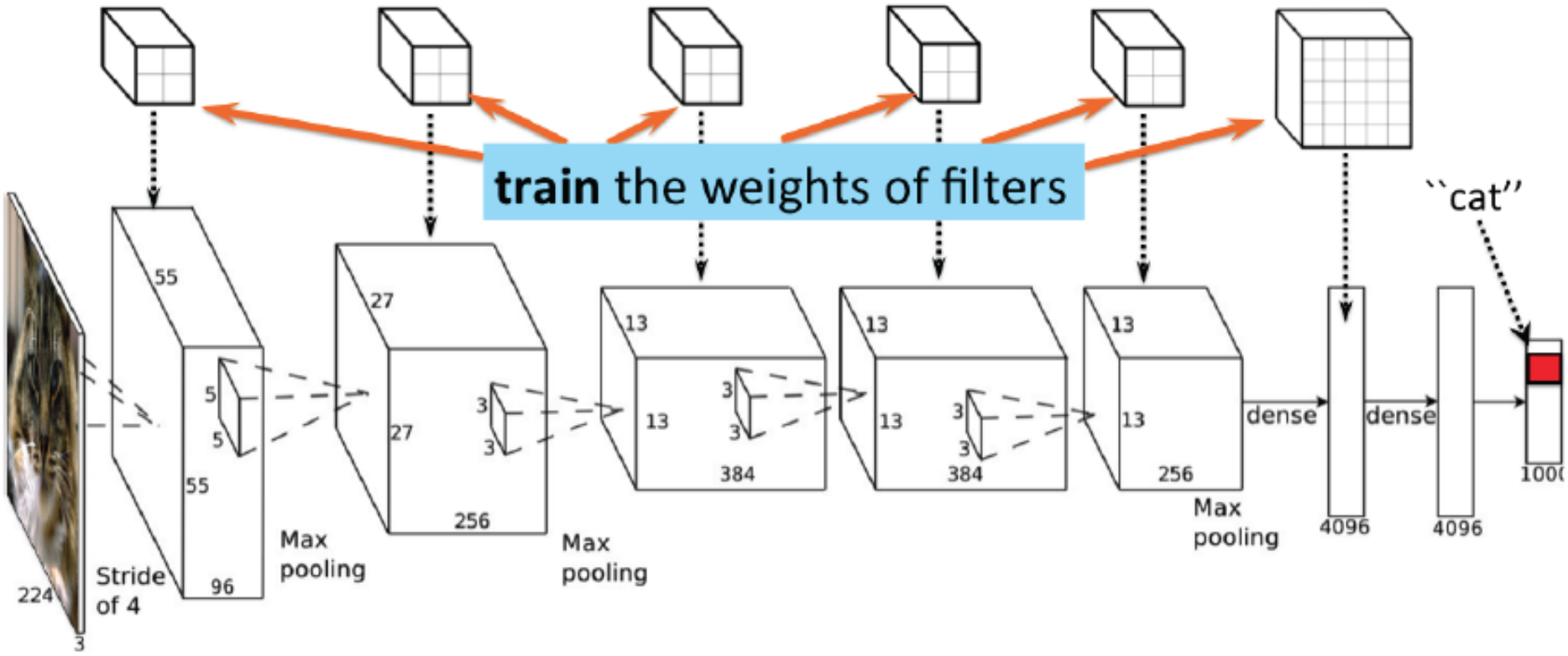
Which filters to use?

Instead of designing them by hand, they are learned using training data.



Which filters to use?

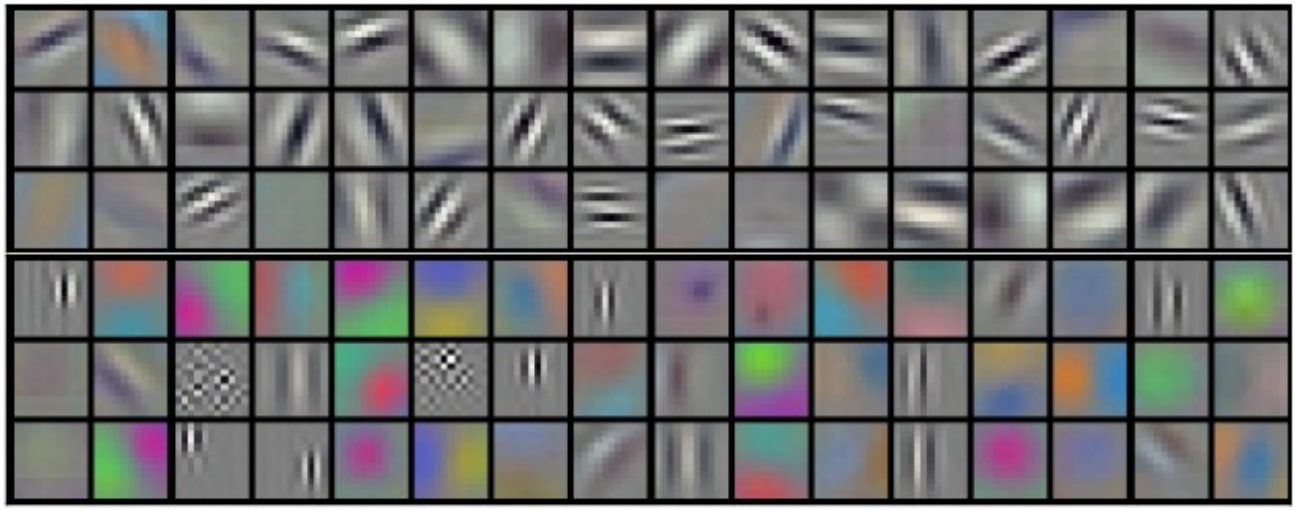
Instead of designing them by hand, they are learned using training data.



Training

Train with MILLIONS of images.

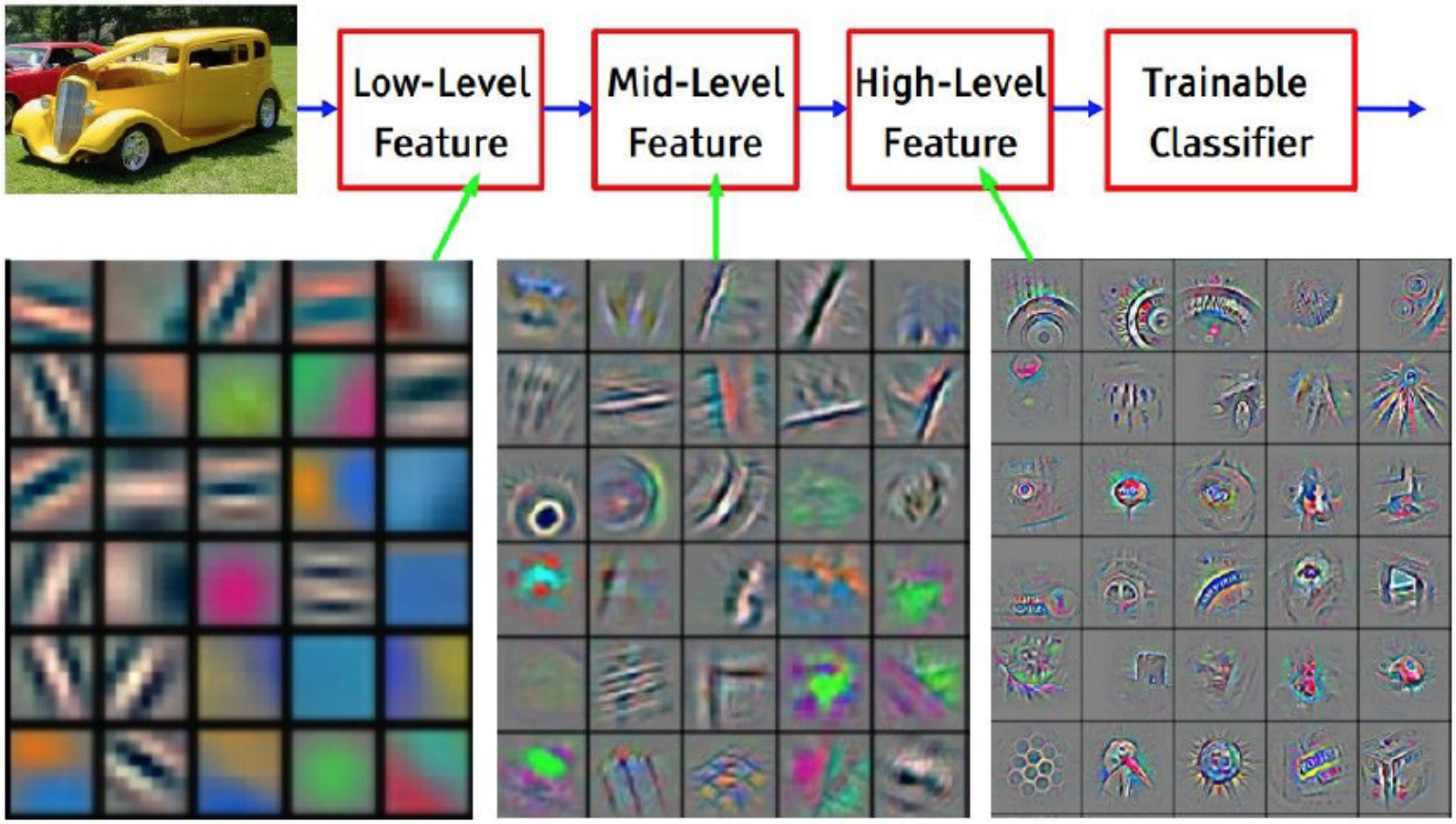
AlexNet First Layer Filters



Feature Visualization

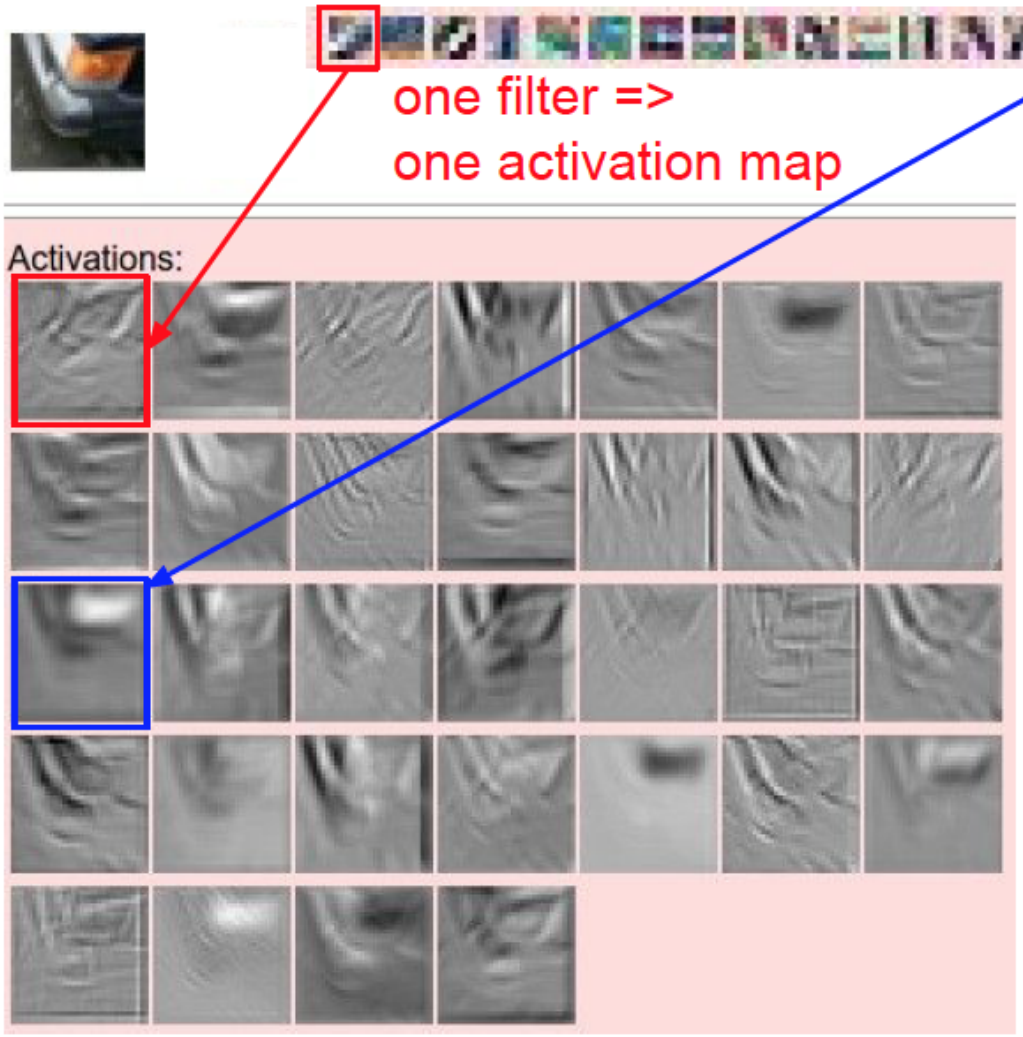
Preview

[From recent Yann LeCun slides]



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Activations



example 5x5 filters
(32 total)

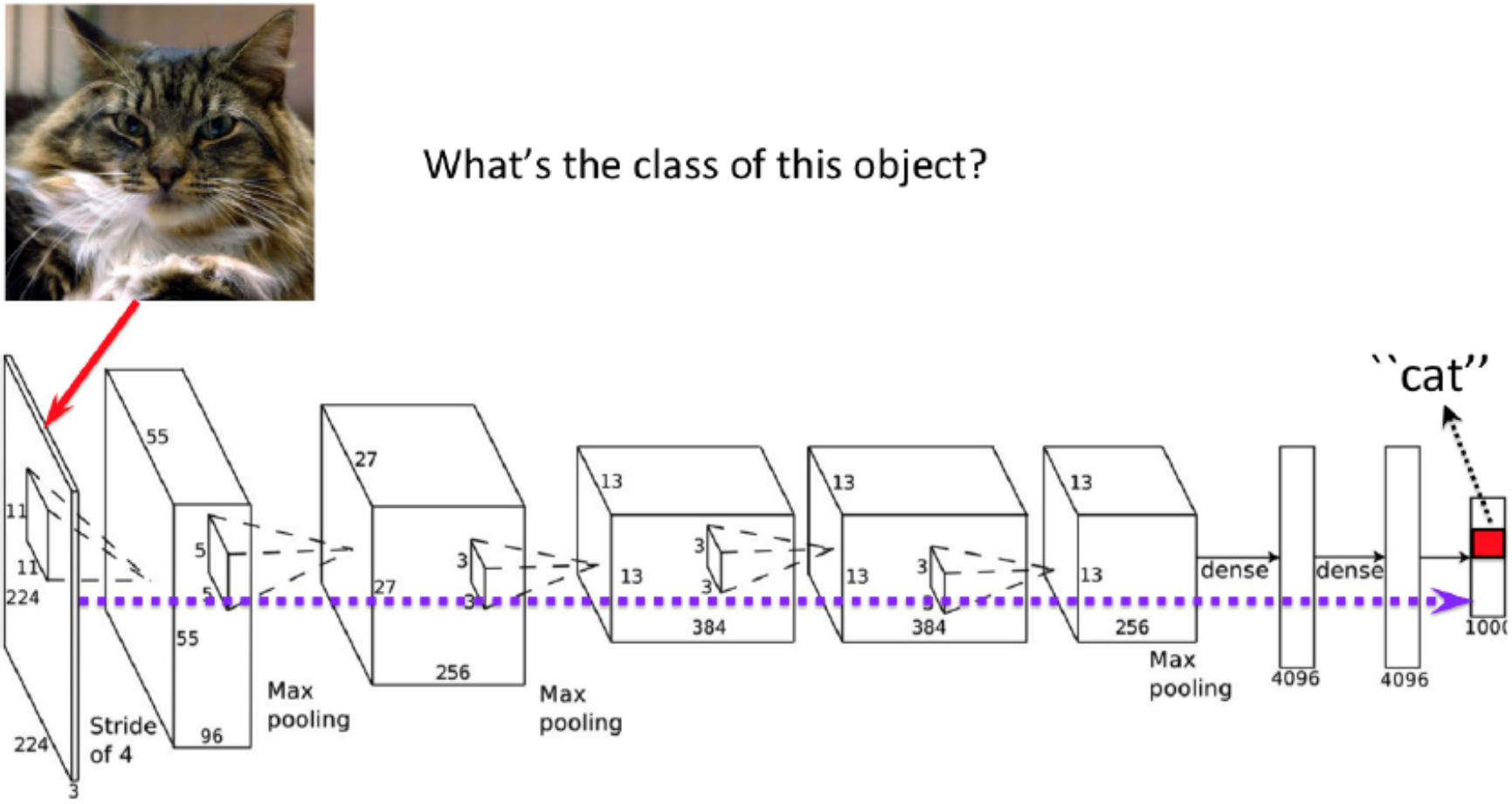
We call the layer convolutional because it is related to convolution of two signals:

$$f[x,y] * g[x,y] = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} f[n_1,n_2] \cdot g[x-n_1, y-n_2]$$

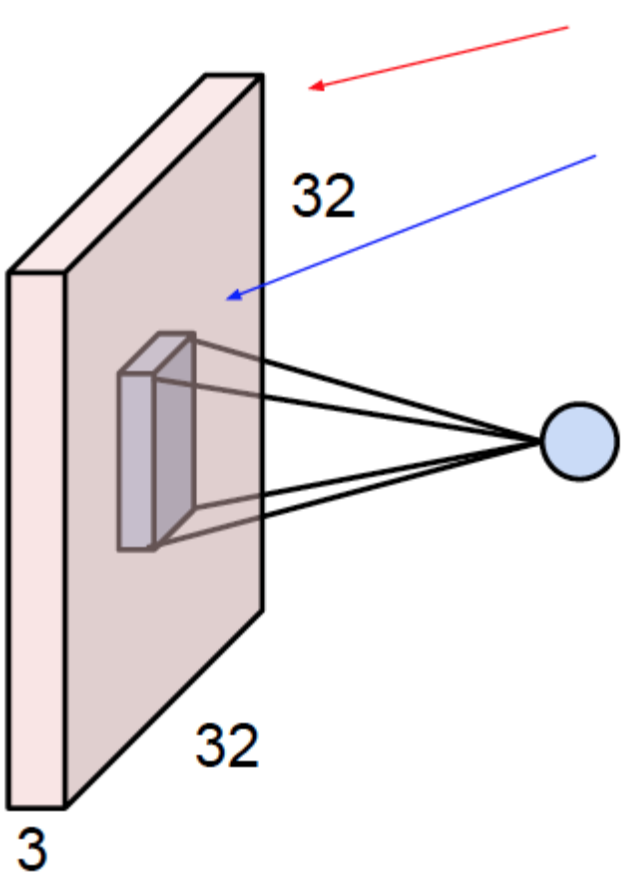
elementwise multiplication and sum of a filter and the signal (image)

Classification

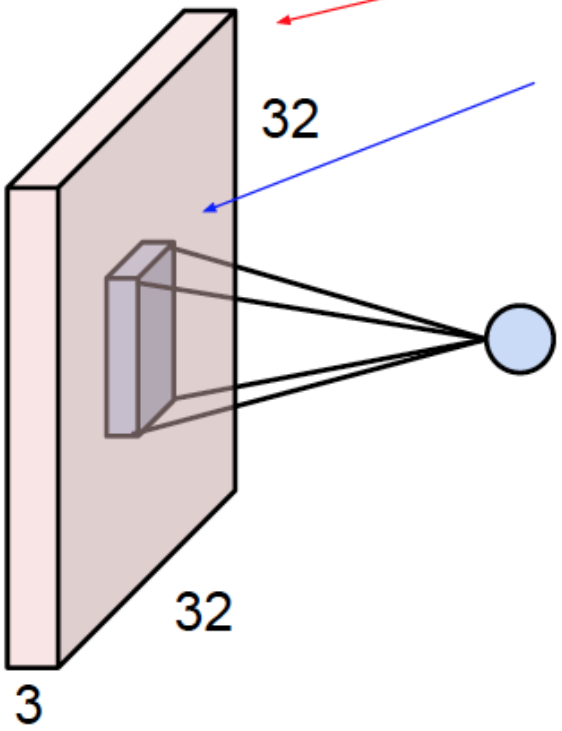
Once trained, the net can be used to classify. Just feed the image or a crop region, run it through the net and read out the class with the highest score in the last layer.



The brain/neuron view of CONV Layer

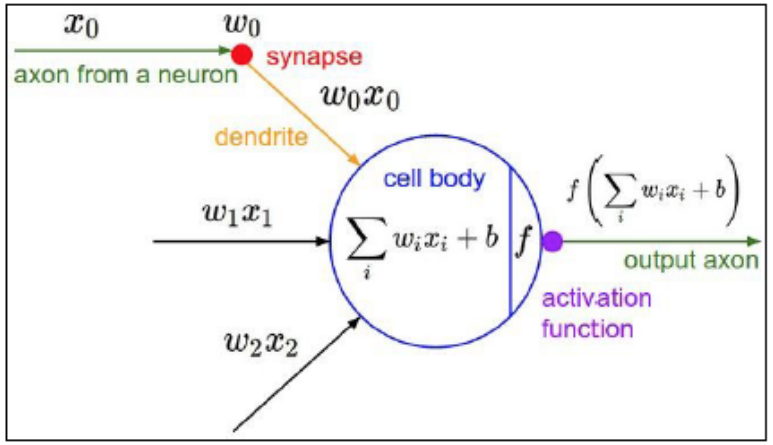


1 number:
the result of taking a dot product between the filter and this part of the image
(i.e. 5*5*3 = 75-dimensional dot product)



32x32x3 image

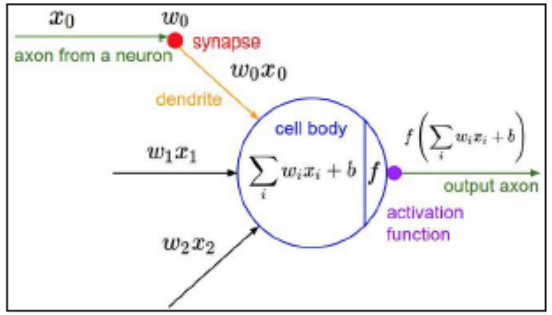
5x5x3 filter

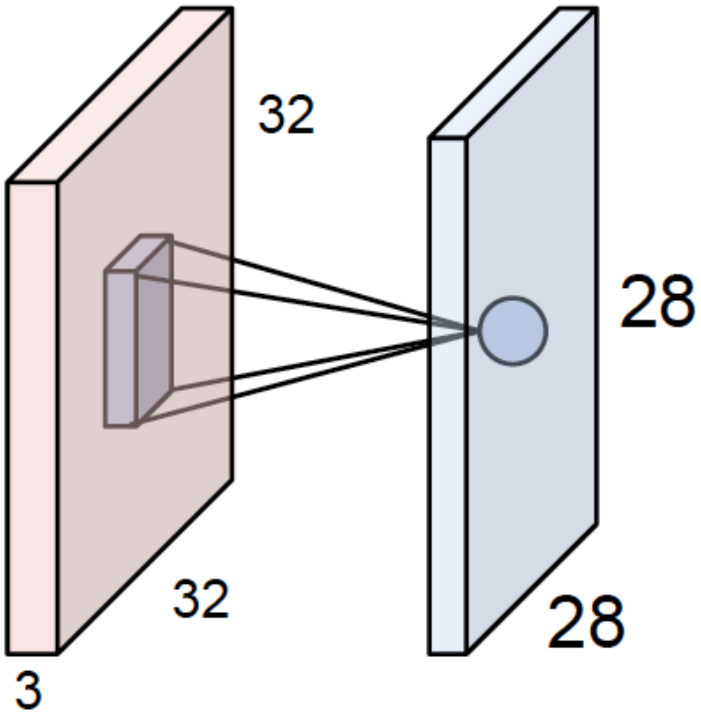


It's just a neuron with local connectivity...

1 number:
the result of taking a dot product between the filter and this part of the image
(i.e. 5*5*3 = 75-dimensional dot product)

The brain/neuron view of CONV Layer



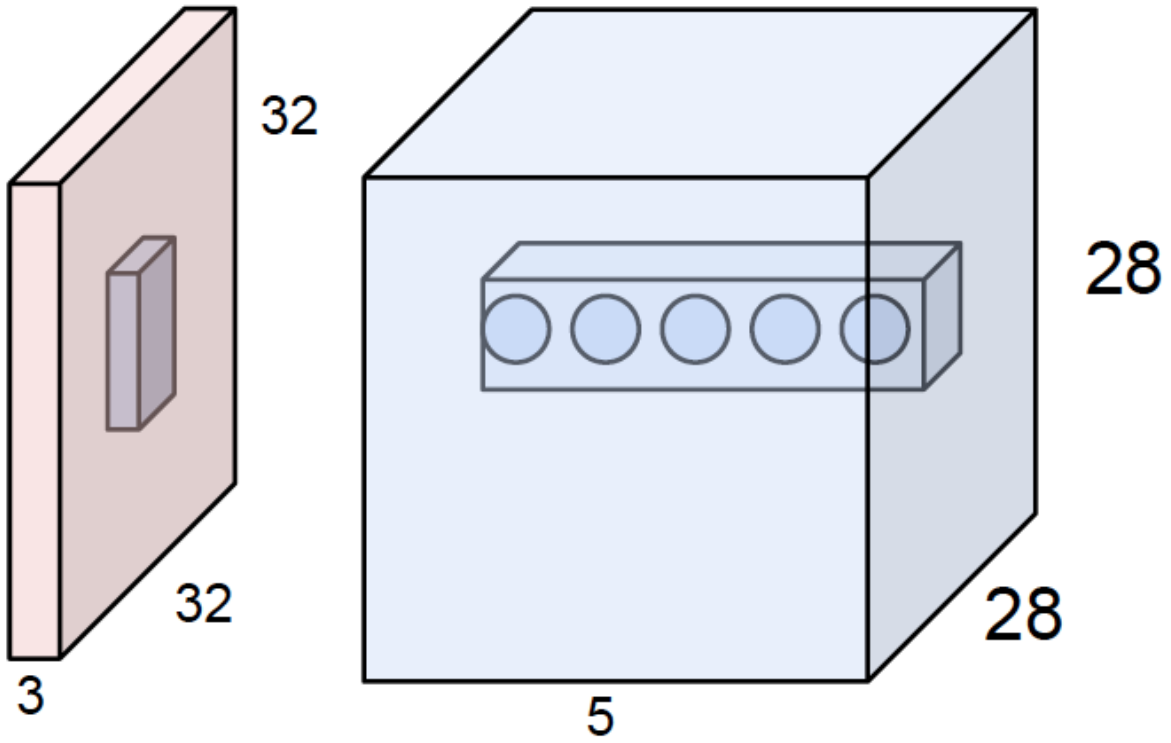


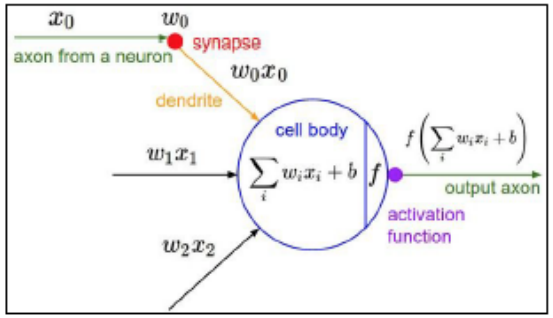
An activation map is a 28x28 sheet of neuron outputs:

1. Each is connected to a small region in the input
2. All of them share parameters

"5x5 filter" -> "5x5 receptive field for each neuron"

The brain/neuron view of CONV Layer



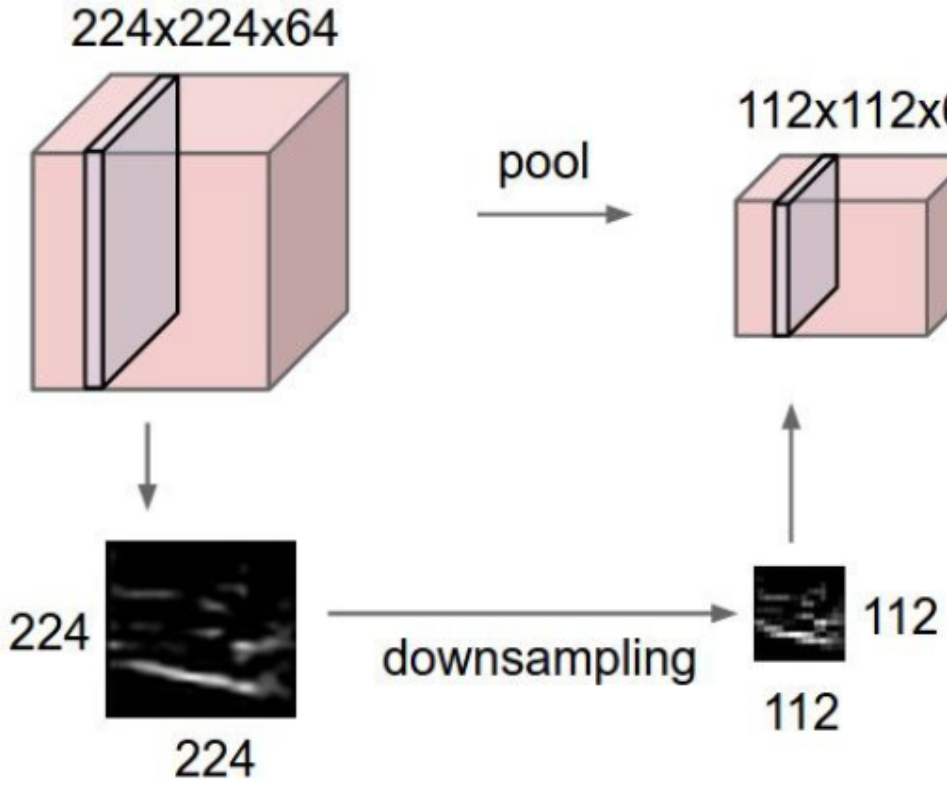


E.g. with 5 filters,
CONV layer consists of neurons arranged in a 3D grid (28x28x5)

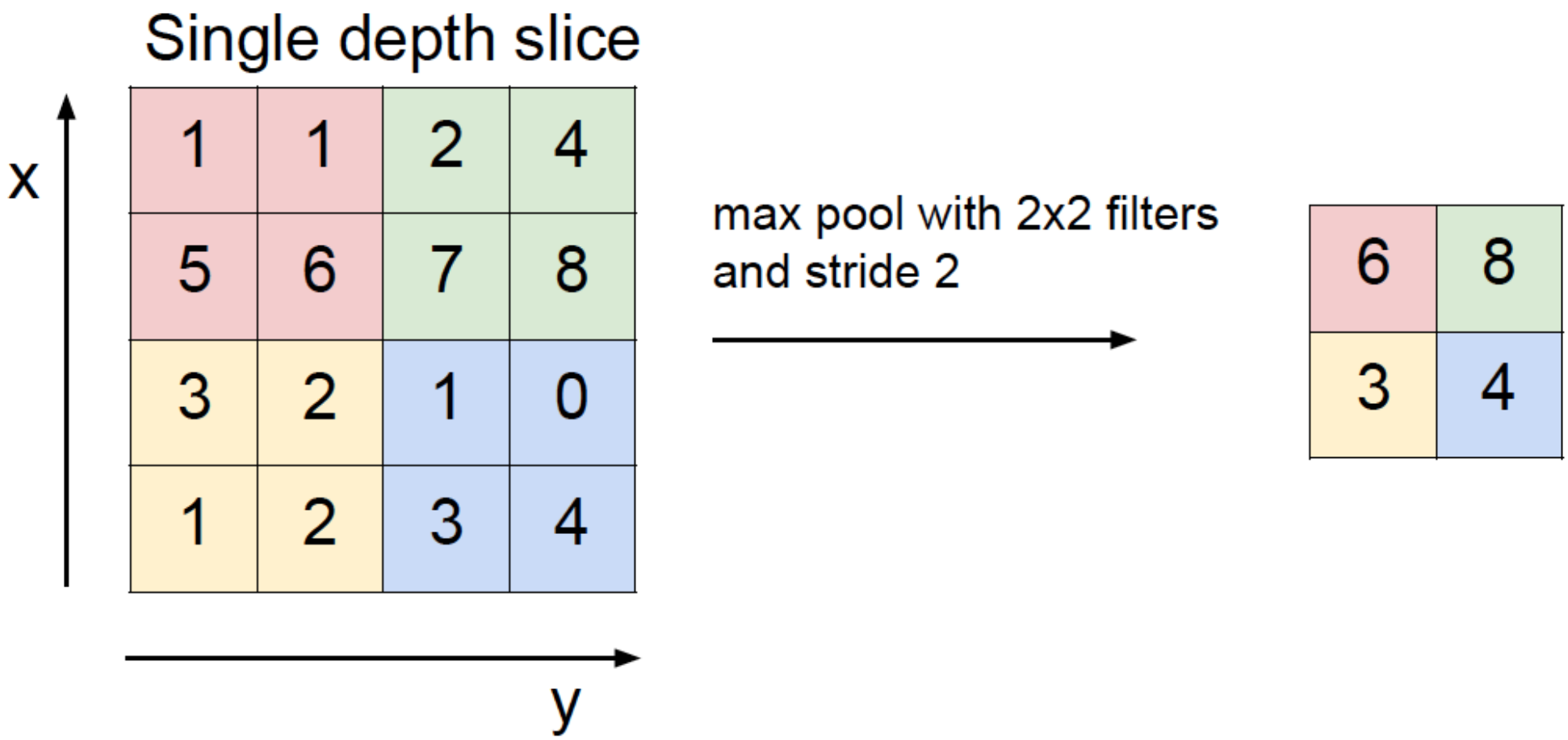
There will be 5 different neurons all looking at the same region in the input volume

Pooling layer

- makes the representations smaller and more manageable
- operates over each activation map independently:

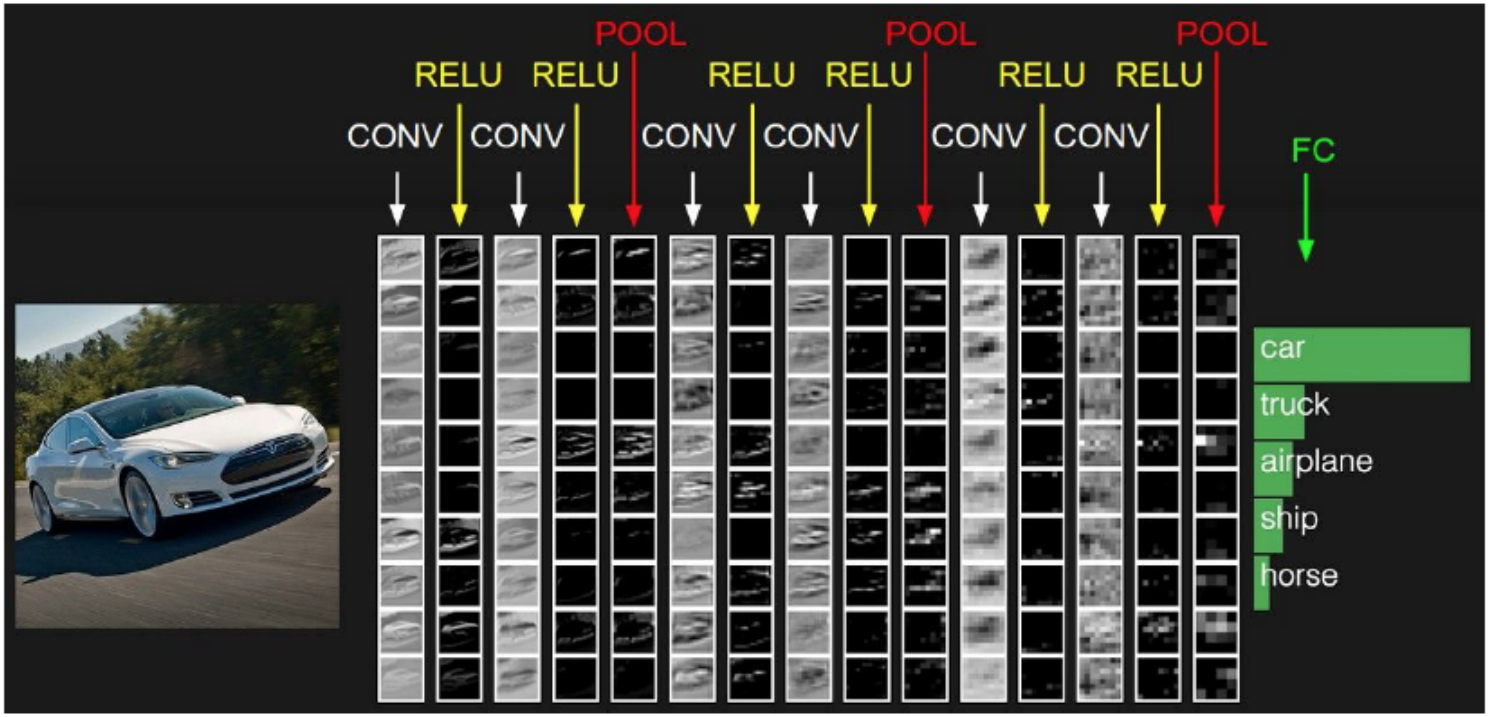


MAX POOLING


Single depth slice
x
y
1 1 2 4
5 6 7 8
3 2 1 0
1 2 3 4
max pool with 2x2 filters
and stride 2
6 8
3 4

Fully Connected Layer (FC layer)

- Contains neurons that connect to the entire input volume, as in ordinary Neural Networks



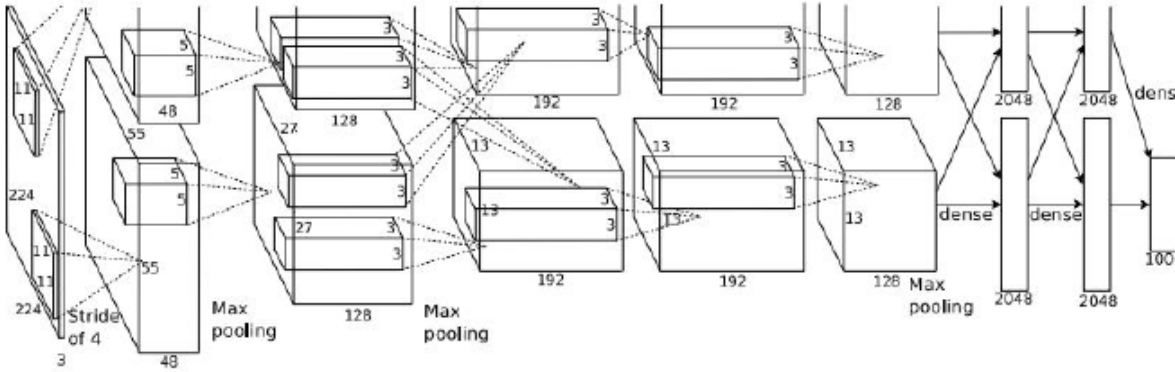
Performance

ImageNet: main challenge for object classification http://image-net.org/

1000 classes, 1.2 M training images, 150K for test

Case Study: AlexNet

[Krizhevsky et al. 2012]



Full (simplified) AlexNet architecture:
[227x227x3] INPUT
[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0
[27x27x96] MAX POOL1: 3x3 filters at stride 2
[27x27x96] NORM1: Normalization layer
[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2
[13x13x256] MAX POOL2: 3x3 filters at stride 2
[13x13x256] NORM2: Normalization layer
[13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1
[13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1
[13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1
[6x6x256] MAX POOL3: 3x3 filters at stride 2
[4096] FC6: 4096 neurons
[4096] FC7: 4096 neurons
[1000] FC8: 1000 neurons (class scores)

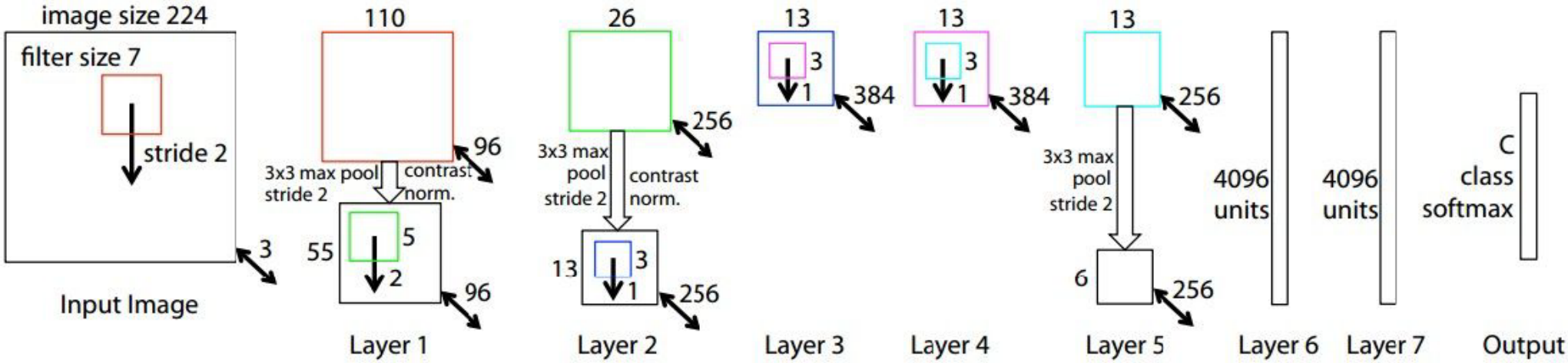
Details/Retrospectives:
- first use of ReLU
- used Norm layers (not common anymore)
- heavy data augmentation
- dropout 0.5
- batch size 128
- SGD Momentum 0.9
- Learning rate 1e-2, reduced by 10 manually when val accuracy plateaus
- L2 weight decay 5e-4
- 7 CNN ensemble: 18.2% -> 15.4%

2012 Performance

A. Krizhevsky, I. Sutskever, and G. E. Hinton won the challenge.

Team name	Filename	Error (5 guesses)	Description
SuperVision	test-preds-141-146.2009-131-137-145-146.2011-145f.	0.15315	Using extra training data from ImageNet Fall 2011 release
SuperVision	test-preds-131-137-145-135-145f.txt	0.16422	Using only supplied training data
ISI	pred_FVs_wLACs_weighted.txt	0.26172	Weighted sum of scores from each classifier with SIFT+FV, LBP+FV, GIST+FV, and CSIFT+FV, respectively.
ISI	pred_FVs_weighted.txt	0.26602	Weighted sum of scores from classifiers using each FV.
ISI	pred_FVs_summed.txt	0.26646	Naive sum of scores from classifiers using each FV.
ISI	pred_FVs_wLACs_summed.txt	0.26952	Naive sum of scores from each classifier with SIFT+FV, LBP+FV, GIST+FV, and

Case Study: ZFNet *[Zeiler and Fergus, 2013]*



AlexNet but:
CONV1: change from (11x11 stride 4) to (7x7 stride 2)
CONV3,4,5: instead of 384, 384, 256 filters use 512, 1024, 512

ImageNet top 5 error: 15.4% -> 14.8%

Case Study: VGGNet

[Simonyan and Zisserman, 2014]

Only 3x3 CONV stride 1, pad 1
and 2x2 MAX POOL stride 2

best model

11.2% top 5 error in ILSVRC 2013
->
7.3% top 5 error

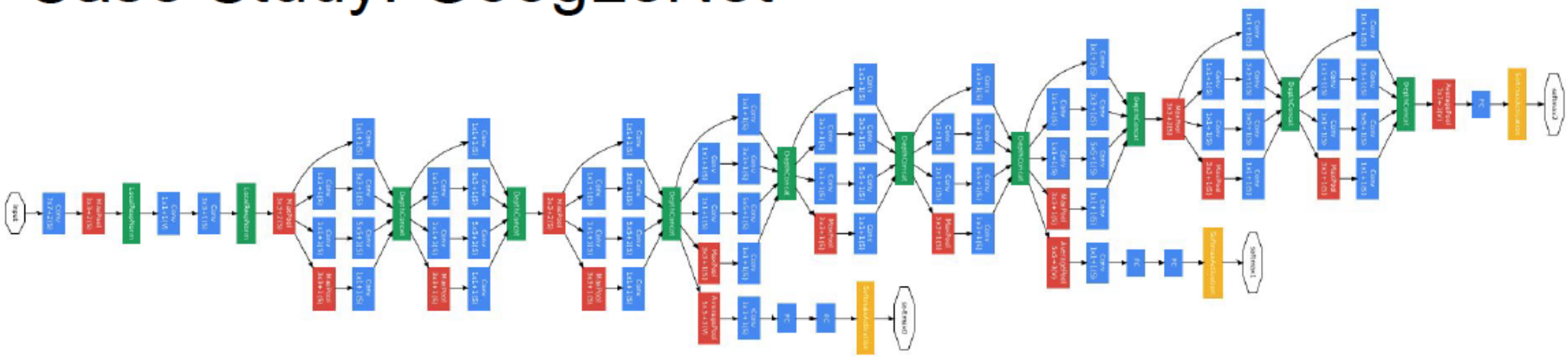
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

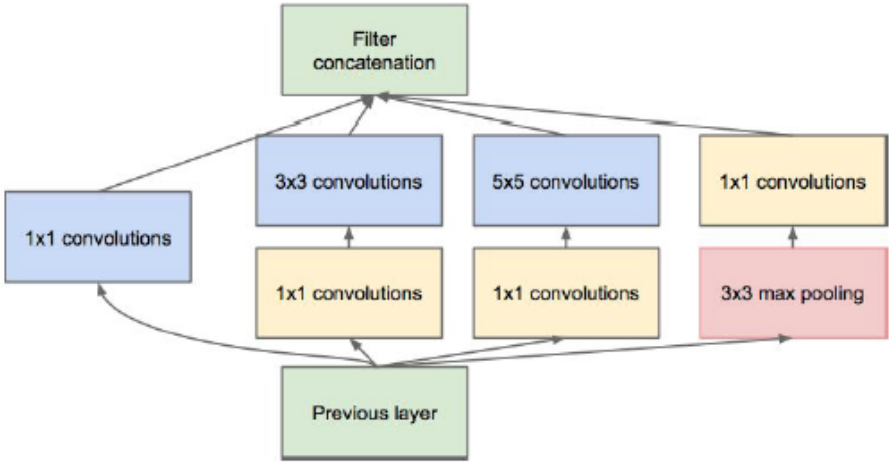
Table 2: **Number of parameters** (in millions).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

Case Study: GoogLeNet

[Szegedy et al., 2014]





Inception module

ILSVRC 2014 winner (6.7% top 5 error)

Case Study: GoogLeNet

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

Fun features:

- Only 5 million params! (Removes FC layers completely)

Compared to AlexNet:
- 12X less params
- 2x more compute
- 6.67% (vs. 16.4%)

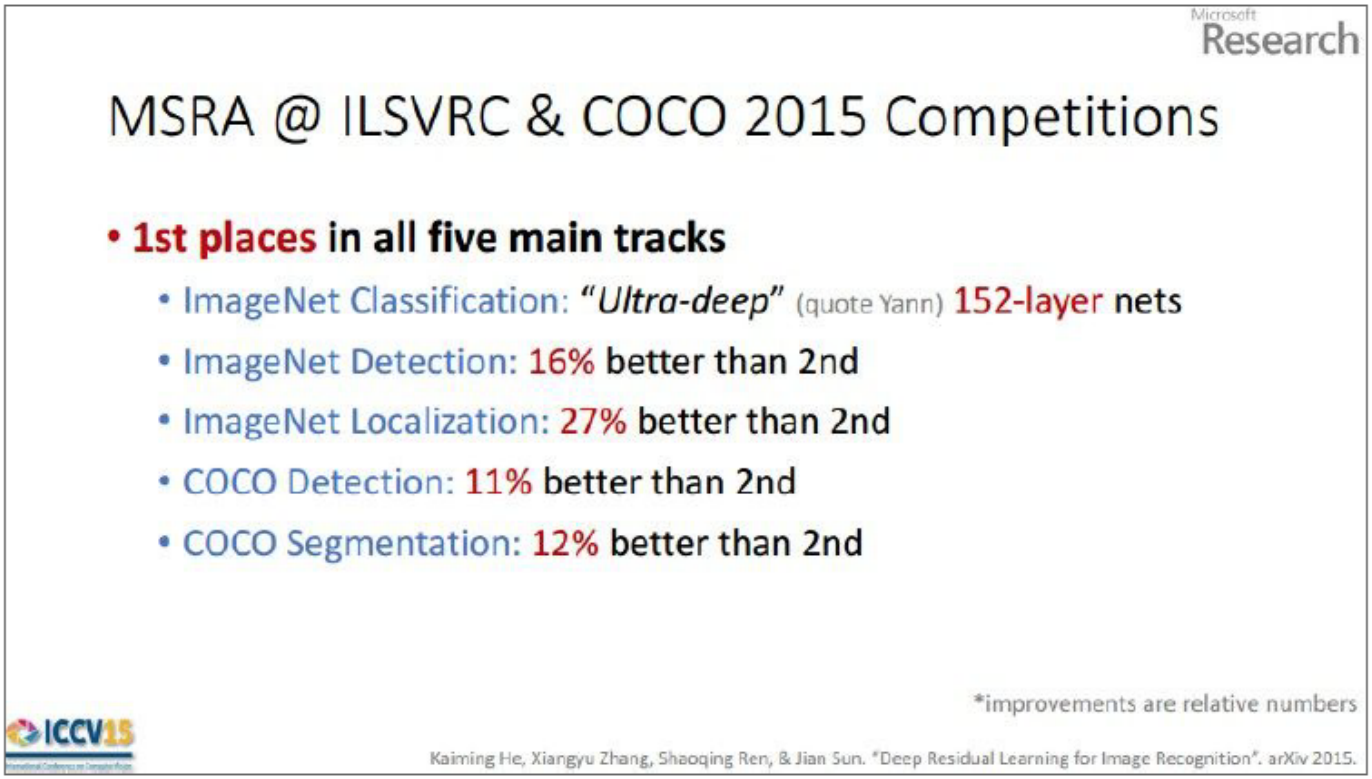
2014 Performance

- Classification / localization error on ImageNet

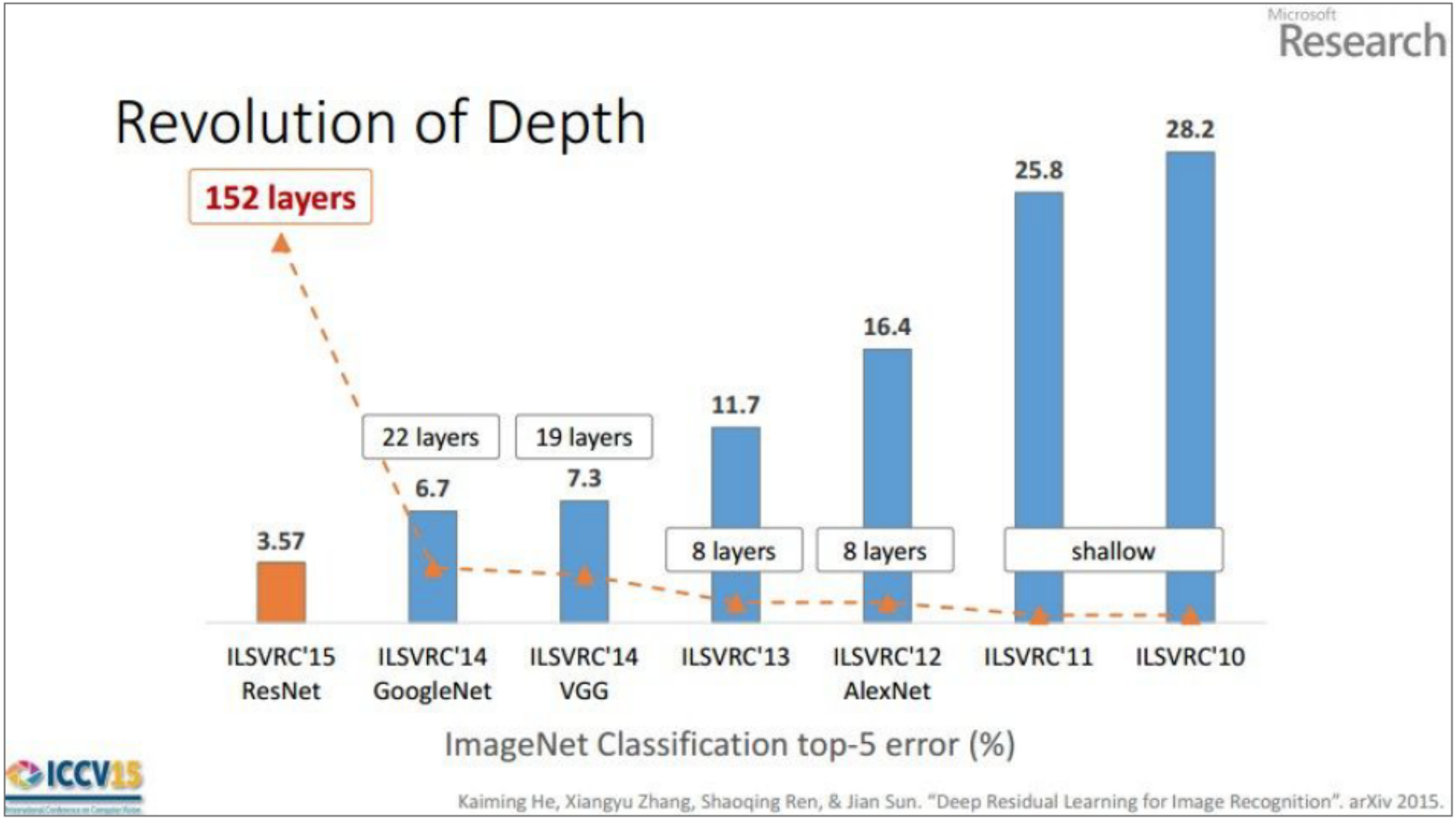
Team name	Entry description	**Classification error**	Localization error
GoogLeNet	No localization. Top5 val score is 6.66% error.	0.06656	0.606257
VGG	a combination of multiple ConvNets, including a net trained on images of different size (fusion weights learnt on the validation set); detected boxes were not updated	0.07325	0.256167
VGG	a combination of multiple ConvNets, including a net trained on images of different size (fusion done by averaging); detected boxes were not updated	0.07337	0.255431
VGG	a combination of multiple ConvNets (by averaging)	0.07405	0.253231
VGG	a combination of multiple ConvNets (fusion weights learnt on the validation set)	0.07407	0.253501
MSRA Visual Computing	Multiple SPP-nets further tuned on validation set (B)	0.0806	0.354924
MSRA Visual Computing	Multiple SPP-nets further tuned on validation set (A)	0.08062	0.354769
Andrew Howard	Combination of Convolutional Nets with Validation set adaptation + KNN	0.08111	0.610365
MSRA Visual Computing	Multiple SPP-nets (B)	0.082	0.355568
Andrew Howard	Combination of Convolutional Nets with Validation set adaptation	0.08226	0.611019
MSRA Visual Computing	Multiple SPP-nets (A)	0.08307	0.3562

Case Study: ResNet *[He et al., 2015]*

ILSVRC 2015 winner (3.6% top 5 error)

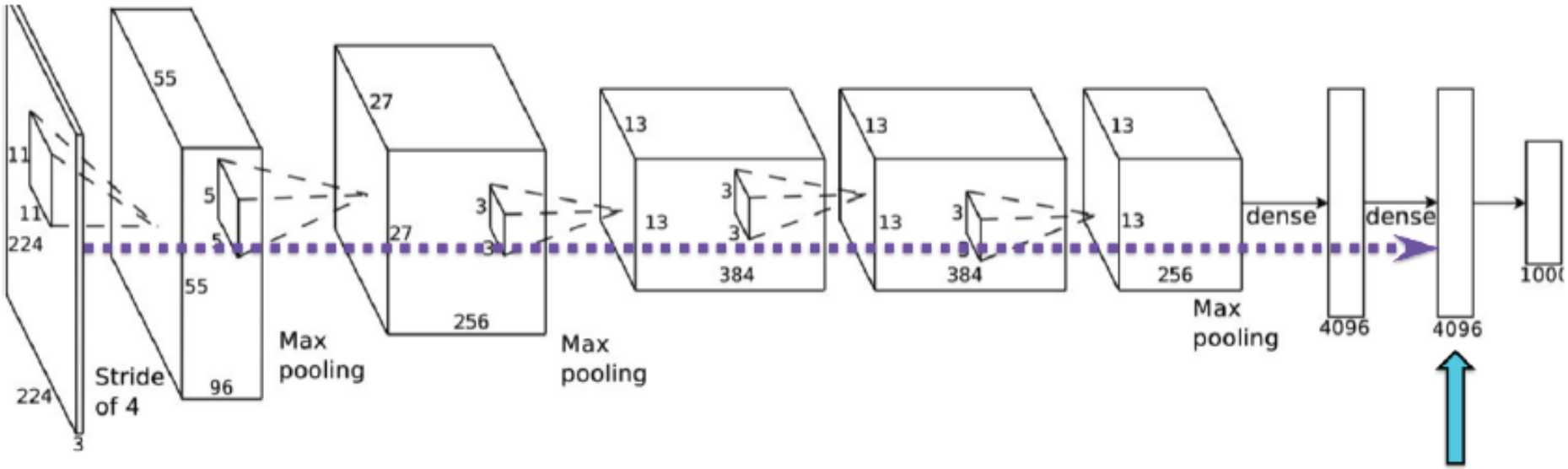


Slide from Kaiming He's recent presentation https://www.youtube.com/watch?v=1PGLj-uKT1w



(slide from Kaiming He's recent presentation)

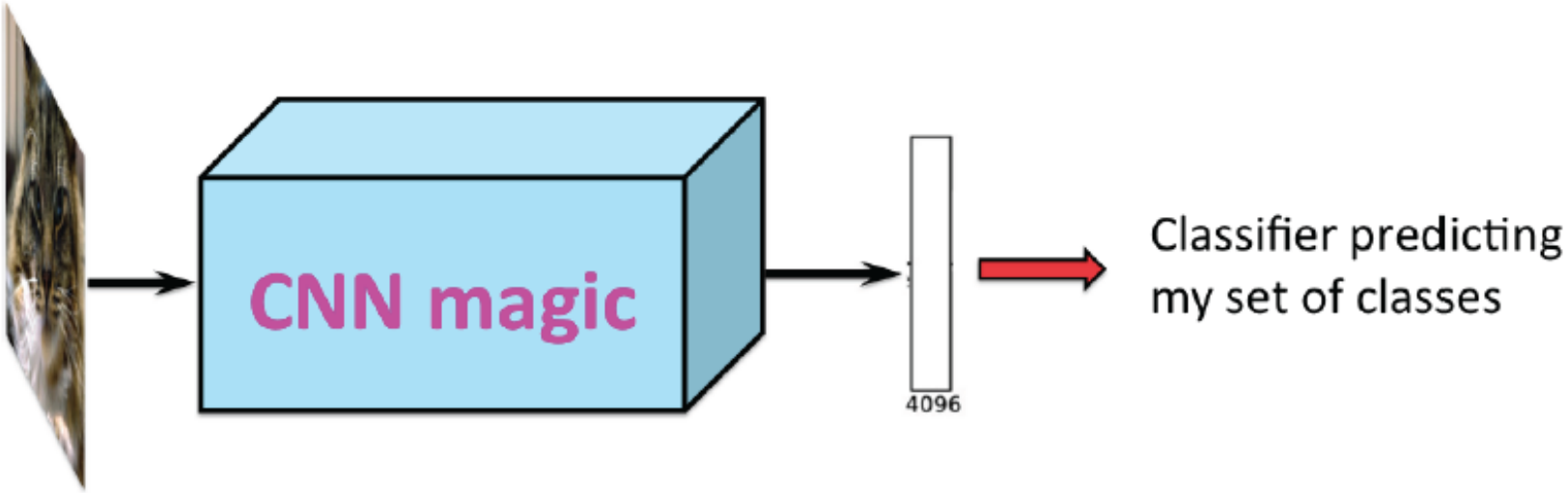
NN as descriptors



Vision people are mainly interested in this vector. **You can use it as a descriptor**. A much better descriptor than SIFT, etc.

Train your own classifier on top for your choice of classes.

NN as descriptors



Detection

Generate LOTS of PROPOSAL bounding boxes

Crop image out of bbox and warp to 224x224

Run through CNN

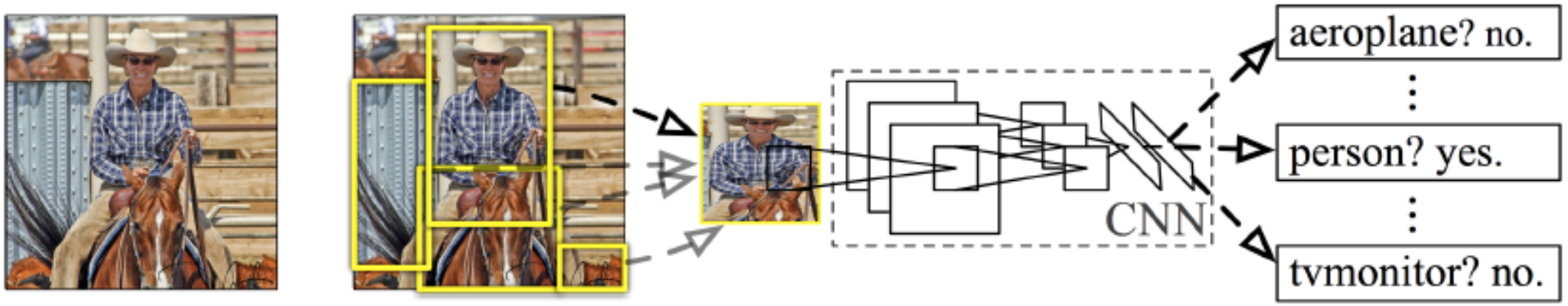


Figure: R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation, CVPR'14

Detection Performance

- **PASCAL VOC challenge**: `http://pascallin.ecs.soton.ac.uk/challenges/VOC/`.

Figure: PASCAL has 20 object classes, 10K images for training, 10K for test

2013 (no NN): 40.4%

	mean	aero plane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motor bike	person	potted plant	sheep	sofa	train	tv/ monitor	submission date
segDPM [?]	40.4	61.4	53.4	25.6	25.2	35.5	51.7	50.6	50.8	19.3	33.8	26.8	40.4	48.3	54.4	47.1	14.8	38.7	35.0	52.8	43.1	24-Feb-2014
Boosted HOG-LBP and multi-context (LC, EGC, HLC) [?]	36.8	53.3	55.3	19.2	21.0	30.0	54.5	46.7	41.2	20.0	31.5	20.8	30.3	48.6	55.3	46.5	10.2	34.4	26.6	50.3	40.3	29-Aug-2010
MITUCLA_Hierarchy [?]	36.0	54.3	48.5	15.7	19.2	29.2	55.6	43.5	41.7	16.9	28.5	26.7	30.9	48.3	55.0	41.7	9.7	35.8	30.8	47.2	40.8	30-Aug-2010
HOGLBP_context_classification_rescore_v2 [?]	34.2	49.1	52.4	17.8	12.0	30.6	53.5	32.8	37.3	17.7	30.6	27.7	29.5	51.9	56.3	44.2	9.6	14.8	27.9	49.5	38.4	30-Aug-2010
LSVM-MDPM [?]	33.7	52.4	54.3	13.0	15.6	35.1	54.2	49.1	31.8	15.5	26.2	13.5	21.5	45.4	51.6	47.5	9.1	35.1	19.4	46.6	38.0	26-Aug-2010
UOCTTI_LSVM_MDPM [?]	33.4	49.2	53.8	13.1	15.3	35.5	53.4	49.7	27.0	17.2	28.8	14.7	17.8	46.4	51.2	47.7	10.8	34.2	20.7	43.8	38.3	21-May-2012
Detection Monkey [?]	32.9	56.7	39.8	16.8	12.2	13.8	44.9	36.9	47.7	12.1	26.9	26.5	37.2	42.1	51.9	25.7	12.1	37.8	33.0	41.5	41.7	30-Aug-2010
RMA2C [?]	32.8	49.8	50.6	15.1	15.5	28.5	51.1	42.2	30.5	17.3	28.3	12.4	26.0	45.6	51.8	41.4	12.6	30.4	26.1	44.0	37.6	29-Oct-2013
UOCTTI_LSVM_MDPM [?]	32.2	48.2	52.2	14.8	13.8	28.7	53.2	44.9	26.0	18.4	24.4	13.7	23.1	45.8	50.5	43.7	9.8	31.1	21.5	44.4	35.7	11-May-2012
GroupLoc [?]	31.9	58.4	39.6	18.0	13.3	11.1	46.4	37.8	43.9	10.3	27.5	20.8	36.0	39.4	48.5	22.9	13.0	36.9	30.5	41.2	41.9	30-Aug-2010
UOCTTI_LSVM_MDPM [?]	29.6	45.6	49.0	11.0	11.6	27.2	50.5	43.1	23.6	17.2	23.2	10.7	20.5	42.5	44.5	41.3	8.7	29.0	18.7	40.0	34.5	21-May-2012
Bonn_FGT_Segm [?]	26.1	52.7	33.7	13.2	11.0	14.2	43.2	31.9	35.6	5.8	25.4	14.4	20.6	38.1	41.7	25.0	5.8	26.3	18.1	37.6	28.1	30-Aug-2010
HOG-LBP + DHOG bag of words, SVM [?]	23.5	40.4	34.7	2.7	8.4	26.0	43.1	33.8	17.2	11.2	14.3	14.5	14.9	31.8	37.3	30.0	6.4	25.2	11.6	30.0	35.7	30-Aug-2010
Svr-Segm [?]	23.4	50.5	24.5	17.1	13.3	10.9	39.5	32.9	36.5	5.6	16.0	6.6	22.3	24.9	29.0	29.8	6.7	28.4	13.3	32.1	27.2	30-Aug-2010
HOG-LBP Linear SVM [?]	22.1	37.9	33.7	2.7	6.5	25.3	37.5	33.1	15.5	10.9	12.3	12.5	13.7	29.7	34.5	33.8	7.2	22.9	9.9	28.9	34.1	29-Aug-2010
HOG+LBP+LTP+PLS2ROOTS [?]	17.5	32.7	29.7	0.8	1.1	19.9	39.4	27.5	8.6	4.5	8.1	6.3	11.0	22.9	34.1	24.6	3.1	24.0	2.0	23.5	27.0	31-Aug-2010
RandomParts [?]	14.2	23.8	31.7	1.2	3.4	11.1	29.7	19.5	14.2	0.8	11.1	7.0	4.7	16.4	31.5	16.0	1.1	15.6	10.2	14.7	21.0	25-Aug-2010
SIFT-GMM-MKL2 [?]	8.3	20.0	14.5	3.8	1.2	0.5	17.6	8.1	28.5	0.1	2.9	3.1	17.5	7.2	18.8	3.3	0.8	2.9	6.3	7.6	1.1	30-Aug-2010
UC3M_Generative_Discriminative [?]	6.3	15.8	5.5	5.6	2.3	0.3	10.2	5.4	12.6	0.5	5.6	4.5	7.7	11.3	12.6	5.3	1.5	2.0	5.9	9.1	3.2	30-Aug-2010
SIFT-GMM-MKL [?]	2.3	10.6	1.6	1.2	0.9	0.1	2.8	1.6	6.7	0.1	2.0	0.4	3.0	2.0	4.4	2.0	0.3	1.1	1.2	2.1	1.9	30-Aug-2010

S. Fidler, R. Mottaghi, A. Yuille, R. Urtasun, Bottom-up Segmentation for Top-down Detection, CVPR'13

2014 (NN): 53.7%

	mean	aero plane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motor bike	person	potted plant	sheep	sofa	train	tv/ monitor	submission date
R-CNN (bbox reg)	53.7	71.8	65.8	53.0	36.8	35.9	59.7	60.0	69.9	27.9	50.6	41.4	70.0	62.0	69.0	58.1	29.5	59.4	39.3	61.2	52.4	2014-Mar-13
R-CNN	50.2	67.1	64.1	46.7	32.0	30.5	56.4	57.2	65.9	27.0	47.3	40.9	66.6	57.8	65.9	53.6	26.7	56.5	38.1	52.8	50.2	2014-Jan-30

Figure: Leading method R-CNN is by Girshick et al.

R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation, CVPR'14

2014 (NN): 62.9%

- Results on the main recognition benchmark, the **PASCAL VOC challenge**

	mean	aero plane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motor bike	person	potted plant	sheep	sofa	train	tv/ monitor	submission date
R-CNN (bbox reg) [?]	62.9	79.3	72.4	63.1	44.0	44.4	64.6	66.3	84.9	38.8	67.3	48.4	82.3	75.0	76.7	65.7	35.8	66.2	54.8	69.1	58.8	27-Oct-2014
R-CNN [?]	59.8	76.5	70.4	58.0	40.2	39.6	61.8	63.7	81.0	36.2	64.5	45.7	80.5	71.9	74.3	60.6	31.5	64.7	52.5	64.6	57.2	27-Oct-2014
Feature Edit [?]	56.4	74.8	69.2	55.7	41.9	36.1	64.7	62.3	69.5	31.3	53.3	43.7	69.9	64.0	71.8	60.5	32.7	63.0	44.1	63.6	56.6	04-Sep-2014
R-CNN (bbox reg) [?]	53.7	71.8	65.8	53.0	36.8	35.9	59.7	60.0	69.9	27.9	50.6	41.4	70.0	62.0	69.0	58.1	29.5	59.4	39.3	61.2	52.4	13-Mar-2014
R-CNN [?]	50.2	67.1	64.1	46.7	32.0	30.5	56.4	57.2	65.9	27.0	47.3	40.9	66.6	57.8	65.9	53.6	26.7	56.5	38.1	52.8	50.2	30-Jan-2014

Figure: Leading method R-CNN is by Girshick et al.

R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation, CVPR'14

Today: 83.8%!!

	mean	aero plane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motor bike	person	potted plant	sheep	sofa	tr
Faster RCNN, ResNet (VOC+COCO) [?]	83.8	92.1	88.4	84.8	75.9	71.4	86.3	87.8	94.2	66.8	89.4	69.2	93.9	91.9	90.9	89.6	67.9	88.2	76.8	9
OHEM+FRCN, VGG16, VOC+COCO [?]	80.1	90.1	87.4	79.9	65.8	66.3	86.1	85.0	92.9	62.4	83.4	69.5	90.6	88.9	88.9	83.6	59.0	82.0	74.7	8
SSD500 VGG16 VOC + COCO [?]	78.7	89.1	85.7	78.9	63.3	57.0	85.3	84.1	92.3	61.2	84.6	63.7	90.3	87.8	88.4	84.7	54.9	84.7	74.2	8
HFM_VGG16 [?]	77.5	88.8	85.1	76.8	64.8	61.4	85.0	84.1	90.0	59.9	82.6	61.9	88.5	85.2	85.6	86.9	56.7	79.5	67.5	8
ION [?]	76.4	87.5	84.7	76.8	63.8	58.3	82.6	79.0	90.9	57.8	82.0	64.7	88.9	86.5	84.7	82.3	51.4	78.2	69.2	8
MNC baseline [?]	75.9	86.4	81.1	76.4	64.3	57.8	81.1	80.3	92.0	55.2	82.6	61.0	89.9	86.4	84.6	85.4	53.1	79.8	66.1	8
Faster RCNN baseline (VOC+COCO) [?]	75.9	87.4	83.6	76.8	62.9	59.6	81.9	82.0	91.3	54.9	82.6	59.0	89.0	85.5	84.7	84.1	52.2	78.9	65.5	8
LocNet [?]	74.8	86.3	83.0	76.1	60.8	54.6	79.9	79.0	90.6	54.3	81.6	62.0	89.0	85.7	85.5	82.8	49.7	76.6	67.5	8